DESIGN OF A SINGLE-DEGREE-OF-FREEDOM BIPED WALKING MECHANISM

Undergraduate Honors Thesis

By

Brett C. Brown

* * * * *

The Ohio State University
Department of Mechanical Engineering
2006

Undergraduate Honors Research
Examination Committee Approval:

Dr. Eric R. Westervelt, Advisor

Dr. James P. Schmiedeler, Advisor

Acknowledgements

I would like to thank my advisors Dr. Eric Westervelt and Dr. Jim Schmiedeler for the opportunity to work on this research project. They provided a great deal of support and input on my work, and I could not have done it without their help. Their ability to work closely with me has had a great effect on the quality of the research experience for me. My research as an undergraduate honors student has strengthened my ability to think critically about engineering and has enhanced my undergraduate experience in Mechanical Engineering as a whole. The effort put forth by Dr. Westervelt and Dr. Schmiedeler as my research advisors is greatly appreciated.

Abstract

Bipedal locomotion is of unique interest due to its implications for human pathology in terms of rehabilitation and prosthetic design. Current biped robots require multiple actuators (motors) because they have multiple degrees of freedom. Furthermore, they require complex control strategies to enable stable walking. These two characteristics have resulted in biped prototypes that are complicated, expensive, heavy, and energetically inefficient. The proposed project attempts to alleviate these problems by creating a single-degree-of-freedom kinematic mechanism that coordinates all of the robot's movement. The findings of this research outline a systematic approach for the design of a kinematic mechanism that has femur and tibia motions that represent those of a stable biped walking gait. A mechanism is designed using this approach, and optimal mechanism parameters are described. The optimized mechanism motions are then compared to those of a biped walking gait known to be stable. It is concluded that a single-degree-of-freedom mechanism is able to reasonably produce the motions of a stable biped walking gait.

Table of Contents

List of Figures

List of Tables

1. Introduction

Objective

The objective of this research is to design a single-degree-of-freedom kinematic mechanism that can accomplish biped walking. The mechanism shall consist of two identical legs. Each leg will have several links. A femur link and a tibia link will make up two of the links of the mechanism. This mechanism will be driven by a single actuator. Other desirable characteristics of this robot are that it be inexpensive, lightweight, and energetically efficient. Furthermore, the robot should move at a reasonable speed. The optimal design should be one in which the resulting walking gaits are stable under surface slope perturbations. One of the most important concepts of this project is the desired simplicity of the biped.

Motivation

In the past 35 years, considerable advancement has been made in the field of bipedal locomotion [1, 2, 3]. The Walking Machine Catalogue [1] gives a list of many of the hundreds of walking robot prototypes that have been created in recent years.. This expansive list of robots includes single-legged machines, as well as machines with up to as many as eight legs. The applications of most of these robots are purely research related. This large list emphasizes the importance and amount of effort put forth to study the field of robotic legged locomotion. A specific example of a two-legged robot is Honda Corporation's ASIMO [2], a 1.2m tall, 52 kg autonomous biped. A picture of this robot is presented in Figure 1.

Figure 1 – Honda's ASIMO Robot [2]

This robot can walk on uneven surfaces, turn smoothly, and climb stairs. ASIMO has 26 motors, each of which drives a degree of freedom (DOF). There are eight sensing units, which are used by an algorithm to drive the 26 DOF's to control walking. It is undeniable that ASIMO is a feat of engineering ingenuity. ASIMO is a very robust robot that can perform a wide range of motions. Yet, the complexity of ASIMO and biped prototypes like it has several downsides. These prototypes are expensive, heavy, and energetically inefficient. When the wide range of motions and robustness of a robot like ASIMO are not necessary, a suitable alternative would be a single-degree-of-freedom mechanism.

One type of robot that contrasts the highly complex ASIMO is a passive walking robot [3]. Unlike a powered robot which uses motors to supply energy to complete the walking motions, the only energy used to drive a passive robot is the robot's potential energy. The gait stability of a passive walking robot is due to the interaction of the robot's mechanics with the environment. Passive robots are extremely limited in their applications. They can only progress over sloped surfaces because the potential energy lost when moving down the slope is converted to the kinetic energy that drives the robot. Passive walkers are certainly an interesting area of study within the field of legged locomotion, but their inadequacies make them impractical for most real-world applications.

The use of a single degree of freedom to drive a mechanism that is not bipedal has been studied before. An example of such a mechanism is Lilly's [4] quadruped trotting machine. Each leg of this quadruped has a single degree of freedom. The motion of each leg is coordinated by the use of cams. Lilly's work demonstrates that a single-degree-of-freedom mechanism can produce suitable motions for quadrupedal trotting.

No biped robot currently exists that integrates all of its motion into one degree of freedom driven by a single actuator. By reducing the number of inputs to one, the motion control of the robot is simplified significantly. The inspiration for this idea comes from [5] and [6]. Other advantages of using a single actuator are a more energetically efficient, less costly prototype. Also, instead of multiple sensing units, like the eight on ASIMO, only one will be needed for the proposed biped. This will also help reduce the biped's complexity and cost.

In order to summarize the pros and cons of a single DOF walking mechanism, Table 1 has been created.

Table 1 - Pros and Cons of a Single DOF Mechanism Compared to a High DOF Robot

PROS	CONS
Only one motor is needed	Range of motions is limited
Fewer sensors are needed	Motions may not be optimal
Control strategy is simplified	Structural integrity of robot may suffer
Cheaper	--
Lighter	--

Needless to say, there are several challenges in creating a biped robot driven by a single actuator. The inherent challenges that exist to ensure stability of walking motions of any biped will certainly be present [7]. A challenge unique to this model will be designing the single-degree-of-freedom mechanism that will provide maximum stability and desired forward progression.

Biped Walking Theory

Biped walking refers to the type of locomotion in which two legs are used and only one leg at a time is off the ground. A single stride includes two distinct phases. These phases are separated by two key events. The two phases are the stance phase and the return phase. With regard to one of the two legs, the stance phase refers to the portion of the walking cycle in which a foot is touching the ground. The return phase refers to the portion of the walking cycle in which a foot is off the ground and returning forward in the direction of progression. While one leg is in the stance phase, the other leg is in the return phase. The two key events are foot liftoff and foot touchdown. Foot liftoff refers to the moment occurring at the end of the stance phase and beginning of the return phase. Foot touchdown refers to the moment that occurs at the end of the return phase and beginning of the stance phase. A step refers to the period of motion between the foot liftoff event and the foot touchdown event of one leg. A stride two steps lasting for the period between foot liftoff and the moment when the same leg returns to the foot liftoff point. Figure 2 below illustrates the gait cycle.

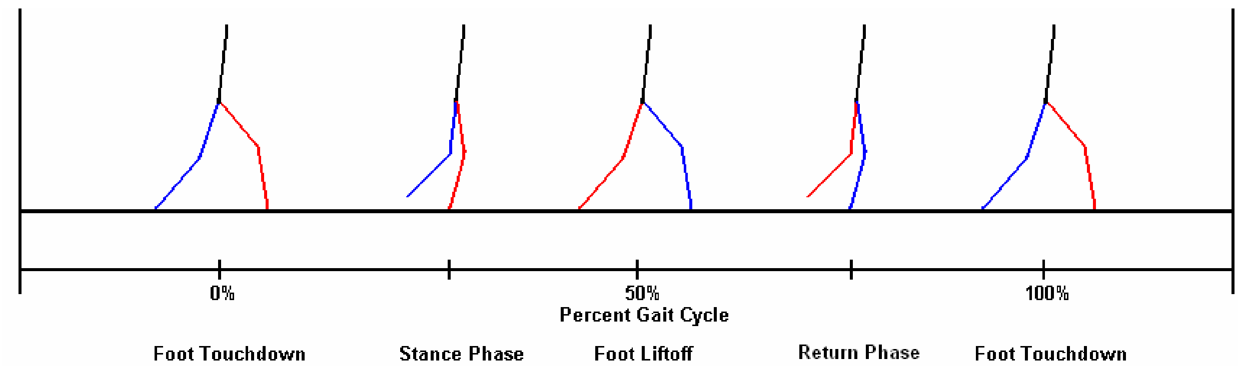

Figure 2 - Biped Walking Gait Cycle

Kinematic Mechanisms Theory

A mechanism is an assembly of rigid bodies, referred to as links, that are connected via joints. Joints restrict the movement of two links relative to one another. The most common joint type is called a revolute joint. Revolute joints allow purely rotational motion along the joint axis. Another type of joint is the prismatic joint, also known as a slider joint. The prismatic joint allows one link to slide in only one coordinate direction relative to another link. There is also a spherical joint. A spherical joint allows rotation of a joint in all three directions relative to the other joint. Revolute and prismatic joints allow one degree of freedom between the links they connect, while a spherical joint allows three degrees of freedom.

The number of degrees of freedom of a mechanism is specifically referred to as the mobility. The mobility of a mechanism is the number of coordinates needed to specify the positions of all members of the mechanism relative to a particular member chosen as the base or frame [8]. Said another way, if a mechanism has a single degree of freedom, the entire configuration of the mechanism is known if one link angle is defined. If a mechanism has two degrees of freedom, the entire configuration of the mechanism is known if two link angles are defined, and so on.

If all the motions of a mechanism are confined to parallel planes, then the mechanism is said to be planar. Planar mechanisms may be visually represented very easily on a 2-D surface since their motion is limited to parallel planes. A typical representation of a planar mechanism consists of solid lines to represent links and open circles to represent a revolute joint, as shown in Figure 3. Furthermore, the frame or base link is identified by a series of dashes extending off the link.

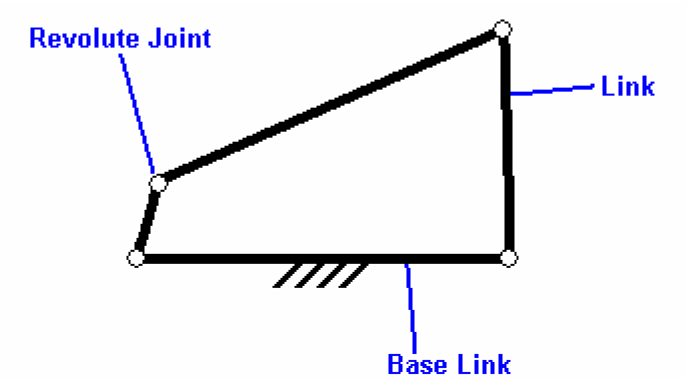

Figure 3 - Basic Mechanism Representation

In order to completely define the motions of a single-degree-of-freedom-mechanism, one must define several variables. First of all, the mechanism configuration must be defined. Second, the lengths of each link of the mechanism must be defined. Finally, the angle of one of the links must be defined. Typically, the angle that is defined is the angle of an input link, which makes a complete rotation.

Outline

In Chapter 2, the procedure used in this research will be explained. It will be stated that first a set of desired motions must be determined. Then, a mechanism configuration must be created which reasonably represents the desired motions. Mechanism design software may be used to assist in this first step. After the mechanism configuration has been determined, the equations of motions of this mechanism must be derived. The process by which one derives the equations of motions of a mechanism is explained. After deriving the equations of motion for the mechanism, it is then possible to optimize the parameters of the equations of motion using the `fmincon` function in MATLAB. The implementation of `fmincon` is discussed in detail. Upon completion of these steps, a mechanism will be designed to produce the desired motions.

In Chapter 3, the results of this research will be discussed. The design process will be summarized and analyzed. The successful implementation of the MATLAB function `fmincon` will be discussed. The result of using a mechanism with more links to represent the femur motion will be analyzed. It is concluded that a mechanism with more links does not necessarily produce motions that more closely match a set of desired motions. The final part of this section will present the final mechanism parameters. A mechanism configuration is defined, and the lengths of each link are also given.

Finally, in Chapter 4, future work that may be extrapolated from this research will be discussed. This research approaches the problem of designing a mechanism to walk as a biped and provides the first part of a solution. There are several more areas of interest that must be studied before it can be concluded that this design is physically feasible and will actually carry out stable walking. These other areas of interest will be discussed. Chapter 4 includes discussion about simulating the final design of the mechanism. It also contemplates analyzing other linkage configurations that may or may not include joint types other than revolute. The idea of force transmission through the mechanism will be discussed. The development of a physical model of a mechanism will be mentioned. Finally, it will be discussed how the methods used in this research to design a biped robot may be applicable to other fields of study as well. A kinematic mechanism could be designed to match any set of desired motions.

2. Design Procedure

Desired Walking Motion

The first question that this research must attempt to answer is what are the desired motions that this mechanism will attempt to produce. By referring to the motions of a robot, one is referring to how the angle of the femur, tibia, and robot's body change as a function of the step progression. A main goal of this research is to provide the process for which to design a mechanism that can replicate a given set of walking motions. This being said, the importance of what motions we choose to be the desired motions becomes less important. However, it is also a goal of this research to lead to the eventual design of a physical mechanism that can perform stable walking. In the end, the walking motions that were chosen to serve as the desired motions come from the robot BIRT [9]. This robot is shown in Figure 4.

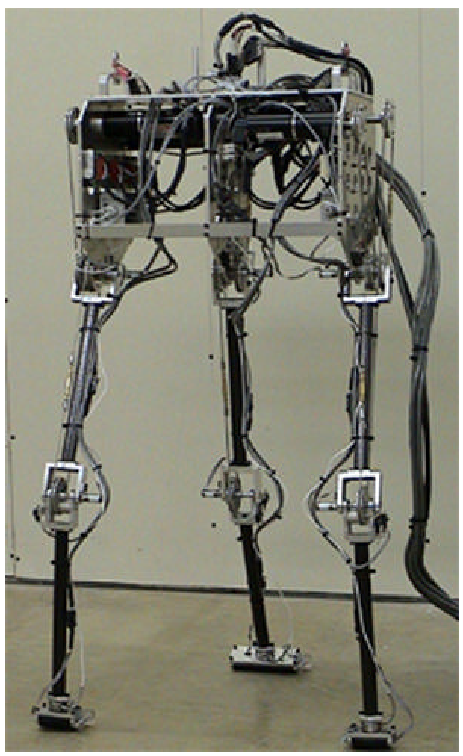

Figure 4 – BIRT [9]

It is noted that BIRT has three legs, yet it is described as a biped robot. The reason BIRT is considered a biped is the outer two legs are slaved together so that they produce the exact same motion. The effect of having two outer legs slaved together is stability in the coronal or side-to-side plane. The main focus of BIRT and the research presented in this thesis is stability in the sagittal plane. This is the plane that extends in the direction of walking. The use of two outer legs has no effect on the stability in the sagittal plane. The motions carried out by BIRT are shown in Figure 5 below.

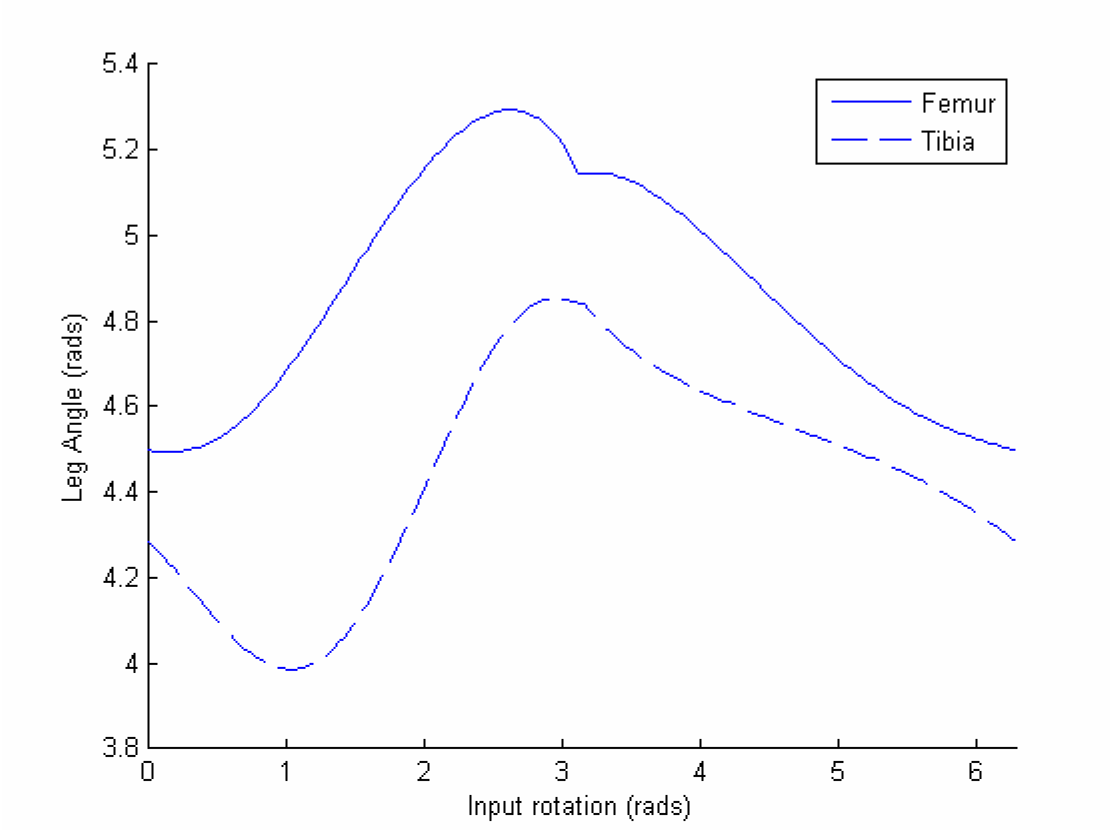

Figure 5 - Femur and Tibia Desired Motions

In Figure 5, the x-axis from 0 to 2*pi radians represents one gait cycle. The motions of the femur and tibia are both plotted here. From zero to pi radians is one step, and from pi radians to 2*pi radians represents the other step that completes the gait cycle. Zero radians represents foot lift-off. From zero to pi radians represents the return phase. Pi radians is the moment of foot touchdown. From pi radians to 2*pi radians represents the stance phase. The y-axis in the figure above is the angle of each leg part measured relative to a fixed reference frame. In this case, the reference was chosen as the horizontal. A discontinuity in the slope of the femur and tibia motion is visibly apparent at pi radians (foot touchdown). Slope discontinuities also exist at 0 radians, although they are less apparent. These slope discontinuities are results of the foot impacting and leaving the ground, respectively. Now that the desired motions have been determined, the process of designing a mechanism to match these motions may begin.

Femur Mechanism Consideration

An infinite number of mechanisms present possibilities for producing the motions of a femur and tibia progressing through a walking motion. In order to aid in the selection of the basic mechanism configuration, the linkage analysis software Robert's Animator [10] has been used. This software helped determine what mechanism configurations produce motions that are conceptually similar to an actual femur and tibia motion during walking. The simplest mechanism to reasonably produce the femur motion was determined to be a four-bar linkage. The mechanism below shows three bars and another bar is considered to be the reference or base link. The base link is the body of the mechanism. An image of the mechanism as seen in Robert's Animator is shown in Figure 6. An important aspect of Robert's Animator is the ability for the user to change the mechanism by clicking and dragging a joint or link and dynamically seeing the change of the output motion as displayed by the solid blue line at the specified joint.

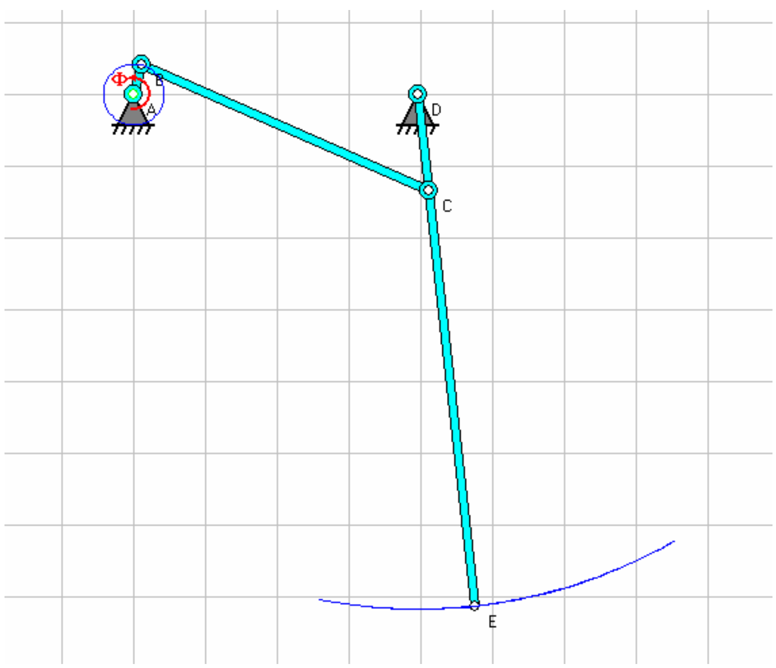

Figure 6 – Femur Four-Bar Linkage Conceptual Design

Femur Four-Bar Mechanism Equations of Motion

In order to analyze the mechanism developed in Robert's Animator, the equations of motion are needed. This means that the angle of the femur link expressed as a function of the input link rotation, the lengths of each link in the mechanism, and the angle of the base link is needed. The procedure [8] for doing this is described here.

The first step in deriving the equations of motion of the mechanism is to express each link as a vector. This is done as shown in Figure 7 below. Notice that the angle of the femur is the same as that of r_4. This analysis finds the angle of r_4.

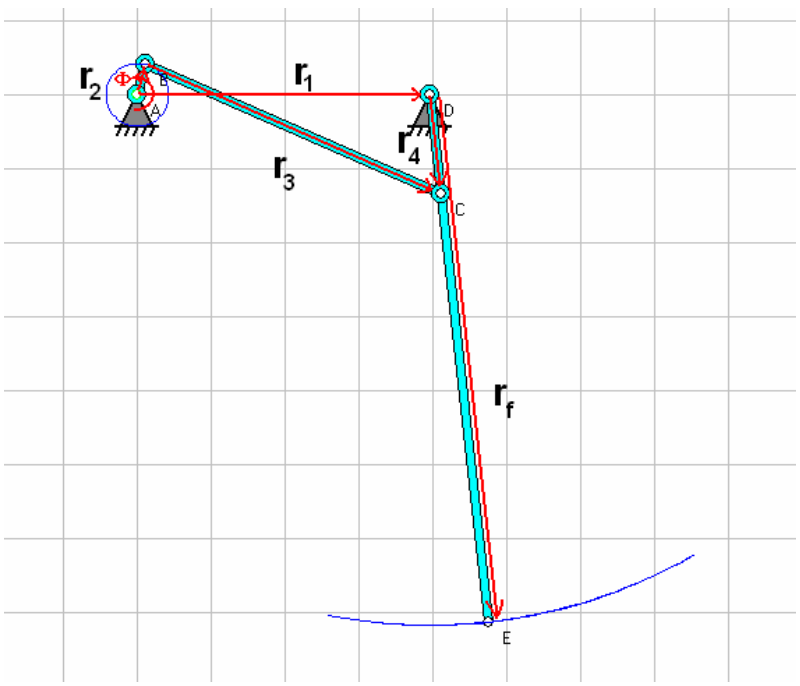

Figure 7 - Femur Four-Bar Mechanism Vectors

Once these vectors are described, a vector loop equation is written. With the vectors described in the figure above, the vector loop equation is formed.

$$\overline{r_1} + \overline{r_4} = \overline{r_2} + \overline{r_3}$$

The $\overline{r_3}$ term is then isolated on the left-hand side, giving the following equation.

$$\overline{r_3} = \overline{r_1} + \overline{r_4} - \overline{r_2}$$

Next, the vectors are described in their x- and y-coordinates as seen in these two equations.

$$r_3 \cos(q_3) = r_1 \cos(q_1) + r_4 \cos(q_4) - r_2 \cos(q_2)$$
$$r_3 \sin(q_3) = r_1 \sin(q_1) + r_4 \sin(q_4) - r_2 \sin(q_2)$$

In order to eliminate the unknown q_3 variable, the two equations are squared and then added together. Completion of this step produces the following equation.

$$r_3 = r_1^2 + r_2^2 + r_4^2 + 2r_1r_4\cos(q_1)\cos(q_4) - 2r_1r_2\cos(q_1)\cos(q_2) - 2r_2r_4\cos(q_2)\cos(q_4) + 2r_1r_4\sin(q_1)\sin(q_4) - 2r_1r_2\sin(q_1)\sin(q_2) - 2r_2r_4\sin(q_2)\sin(q_4)$$

Now, let $A\cos(q_4) + B\sin(q_4) + C = 0$, where *A*, *B*, and *C* are as follows.

$$A = 2r_1r_4\cos(q_1) - 2r_2r_4\cos(q_2)$$
$$B = 2r_1r_4\sin(q_1) - 2r_2r_4\sin(q_2)$$
$$C = r_1^2 + r_2^2 + r_4^2 - r_3^2 - 2r_1r_2\left(\cos(q_1)\cos(q_2) + \sin(q_1)\sin(q_2)\right)$$

Now *t* is defined as follows.

$$t = \frac{-B + s\sqrt{B^2 - C^2 + A^2}}{C - A}$$

The value of s can either be +1 or -1, so there are two solutions for t. Examination of this statement is in the following paragraph. Now, q_4 is calculated.

$$q_4 = 2\tan^{-1}(t)$$

Maple was used to perform the outlined procedure and then find the expression for q_4. The final expression for q_4 is

```
theta4 = -2*atan((-
2*r1*r4*sin(theta1)+2*r2*r4*sin(theta2)+sqrt(4*r1^2*r2^2*cos(theta2)^2+4*r1^2
*r2^2*cos(theta1)^2-4*r3^2*r1*r2*cos(theta1)*cos(theta2)-
4*r3^2*r1*r2*sin(theta1)*sin(theta2)-
8*r1^2*r2^2*cos(theta1)*cos(theta2)*sin(theta1)*sin(theta2)+4*r1^3*r2*sin(the
ta1)*sin(theta2)+4*r2^3*r1*cos(theta1)*cos(theta2)+4*r2^3*r1*sin(theta1)*sin(
theta2)-8*r1^2*r2^2*cos(theta1)^2*cos(theta2)^2-
4*r1*r4^2*sin(theta1)*r2*sin(theta2)-r1^4-r2^4-r4^4-
r3^4+4*r1^3*r2*cos(theta1)*cos(theta2)-
6*r1^2*r2^2+2*r1^2*r4^2+2*r1^2*r3^2+2*r2^2*r4^2+2*r2^2*r3^2+2*r4^2*r3^2-
4*r4^2*r1*r2*cos(theta1)*cos(theta2)))/(-r1^2-r2^2-
r4^2+r3^2+2*r1*r2*cos(theta1)*cos(theta2)+2*r1*r2*sin(theta1)*sin(theta2)+2*r
1*r4*cos(theta1)-2*r2*r4*cos(theta2)));
```

It is noted that the previous equation for t has two solutions. One is for s = +1, and the other is for s = -1. These two different solutions represent two different assembly modes of the mechanism. This can also be stated as there being two distinct configurations possible using the four links. In order to determine which configuration was desired for the purpose of this research, the equations of motion for each case were solved, and then qualitatively observed via a plot. One linkage configuration created an assembly similar to that shown in Figure 6 and Figure 7. This configuration corresponded to $s = -1$. This was the desirable configuration. The other linkage configuration created an assembly distinctly different than the configuration shown in Figure 6 and Figure 7.

Femur Four-Bar Mechanism Optimization

Now there is an equation that describes the femur motion as a function of each link length and the input link rotation. The desired motions of the femur will be supplied by a set of motions performed by the biped robot BIRT. These desired motions have been experimentally shown to provide a stable walking gait. The `fmincon` function in MATLAB is used to optimize the mechanism parameters. The complete MATLAB script file used for the `fmincon` optimization can be found in Appendix B. In order to use the `fmincon` function, the user must first supply an objective function. This objective function must be a function of a value or values that are being optimized. For the purpose of this research, the objective function in the femur optimization was defined as,

$$f = S\ (\ (\ ?_f - \text{femurangle\_desired}\ )^2\ )$$

In this equation, $?_f$ is the function for the angle of the femur in the mechanism. It is parameterized by the length of the four links of the femur mechanism, the input link rotation, and the angle of the r_1 vector which is the ground link. These are the parameters that are being optimized. In other words, this script file will find the values of r_1, r_2, r_3, r_4, and theta1 such that the value of f is minimized. $?_2$ is not optimized because it is already defined. It is not adjustable. The variable femurangle_desired is a vector the same length as $?_f$ that describes the desired angle of the femur for one complete gait cycle. For all of the optimization performed in this research, it was determined that 200 values, spaced equally with respect to the input link rotation, $?_2$, would be adequate to specify the motion. This means that the $?_f$ value and the femurangle_desired value are each vectors of length 200.

In using the `fmincon` function, it is also required that the user supply an initial guess for the parameters being optimized as well as a lower and an upper bound for these values. For any parameter, if the initial guess supplied is not within a certain distance from the actual optimized value, that parameter will not converge to its optimal value, and the script file will not finish running. For this reason, several iterations of running the optimization may be necessary, and it may be necessary to only optimize one parameter at a time at first, using an iterative approach to change the initial guess to a more suitable value. Adjusting the upper and lower limits of the parameters is useful in making sure that the resulting mechanism is physically feasible, meaning that one link is not extremely large and another link extremely small.

Figure 8 below shows a plot of the desired motion and the motion that is actually produced with the optimized link lengths.

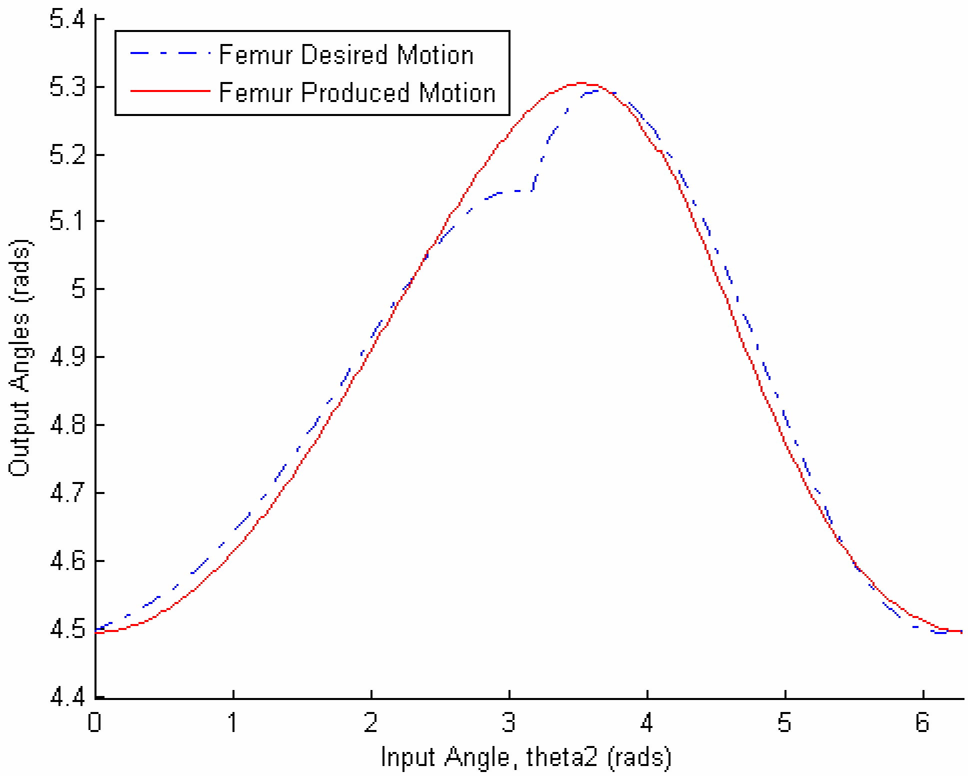

Figure 8 - Desired and Produced Femur Motion

The parameters of the mechanism corresponding to Figure 8 are listed in Table 2. The links represented by the parameter name can be found in Figure 7. It is worth noting that the value of r_1 was chosen to be 1.000 prior to optimization. As mentioned previously, kinematic mechanisms are completely scalable. The rest of the link lengths are determined in proportion to one link length that is defined, in this case r_1. Also, recall that the angle of the femur will be equal to that of the r_4 link regardless of the length of the femur. This is the reason why the length of the femur is not listed in Table 2. The femur length has no effect on how well this mechanism can match the desired femur motion.

Table 2 - Optimized Four-Bar Femur Mechanism Parameters

Parameter	Value
r_1	1.000
r_2	0.269
r_3	1.353
r_4	0.925

As seen in Figure 8, the optimized motion matches the desired motion fairly well. There is one part of the motion that is not closely matched, and that occurs at pi radians in Figure 8. This is the spot where foot touchdown occurs. Foot touchdown is a considerably important event in a stride, so it may be fairly significant that there is an error at this location. This could negatively affect the stability of the gait, and more discussion regarding how to improve this motion will follow in later sections. Now that proper link lengths for the four-bar linkage controlling the femur have been determined, the mechanism to control the tibia, which will be an extension of the femur mechanism, may be analyzed.

Tibia Mechanism Consideration

Now that an adequate mechanism has been determined to drive the femur motion, the same process may be followed to find a suitable mechanism to drive the tibia. Using Robert's Animator software, a mechanism that could reasonably replicate the desired motion of the tibia can be found. The mechanism that is shown in Figure 9. The addition of the links that produce the femur motion make the complete leg mechanism a six-bar linkage.

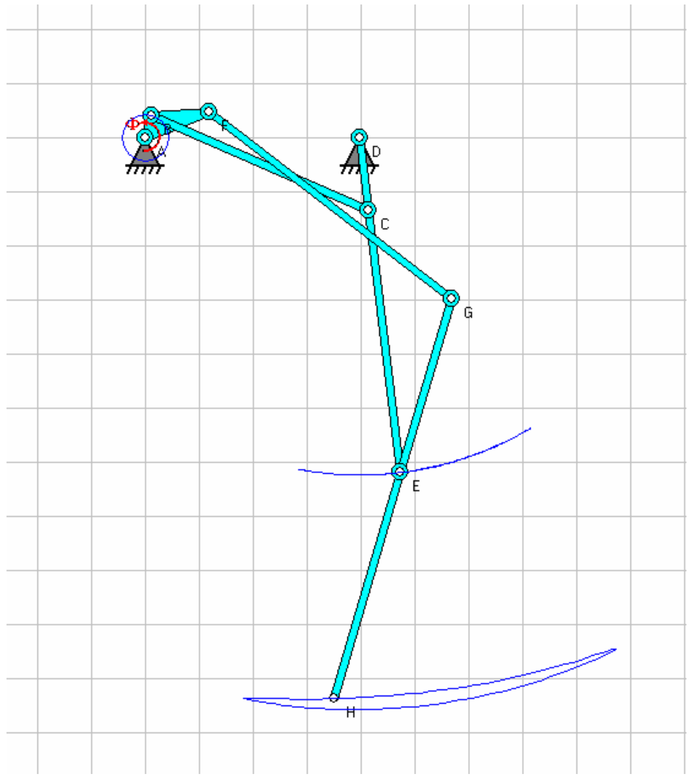

Figure 9 - Tibia Linkage Conceptual Design

An important part of the tibia motion is that it provides clearance for the foot to return forward without hitting the ground. The blue line in Figure 9 shows the path that the foot, point H, follows as the mechanism goes through its motions. It can be seen from this path that the motion produced by the proposed mechanism does indeed provide clearance in the return phase.

Tibia Equations of Motion

The equations of motion for the mechanism that drives the tibia are found using the same method discussed earlier. The vectors chosen to represent the links are shown in Figure 10.

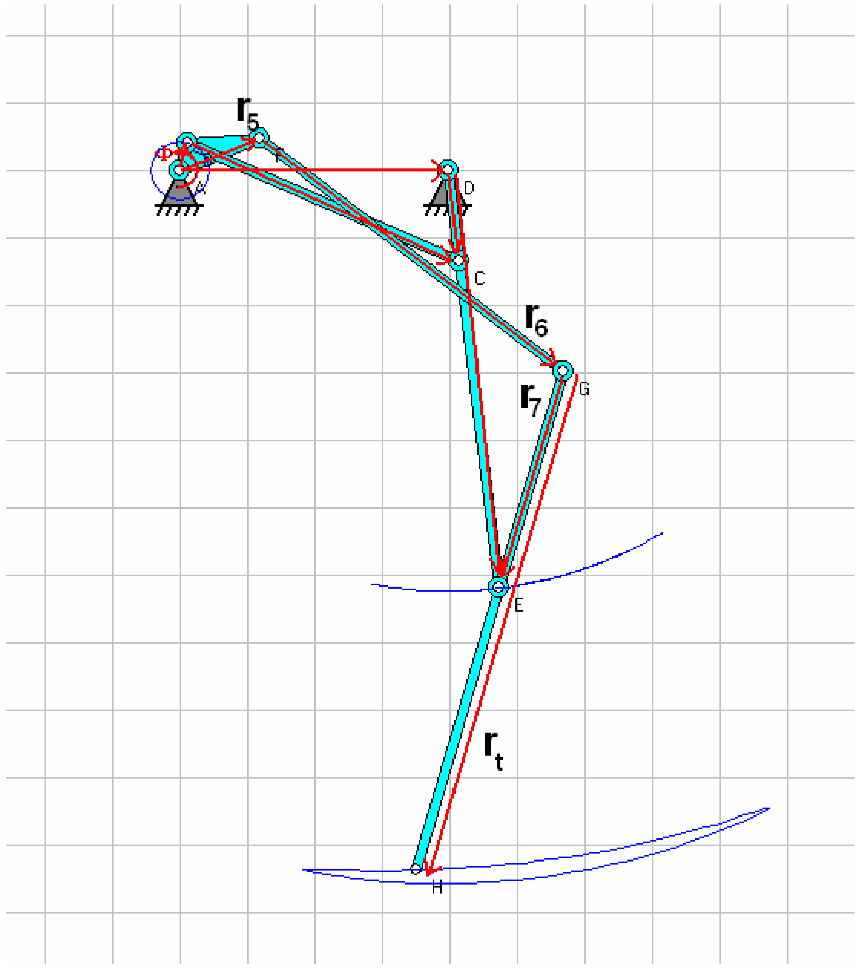

Figure 10 - Tibia Mechanism Vectors

The vector loop equation used to solve the tibia equations of motion utilizes some of the same vectors from the part of the mechanism that controls the femur. The vector loop equation is as follows.

$$\overline{r_6} = \overline{r_1} + \overline{r_f} - \overline{r_5} - \overline{r_7}$$

This equation is expressed in its x and y components as before, each component equation is squared, and they are added together. Then, the equation $A\cos(q_7) + B\sin(q_7) + C = 0$ is defined, and t is expressed as $t = \dfrac{-B \pm \sqrt{B^2 - C^2 + A^2}}{C - A}$. It is noted that the coefficients A, B, and C have different values than for the case of the four-bar femur mechanism. The A, B, and C coefficients must be determined with regards to the tibia mechanism vectors. Finally, q_7 is found from the equation $q_7 = 2\tan^{-1}(t)$. The angle of the tibia is then known because it is equal to the angle of link 7. Again, there are two possible values for t relating to the two different assembly modes of the mechanism. The correct assembly mode can be determined by creating a plot of the mechanism and comparing that to the desired assembly configuration as shown in Figure 9 and Figure 10. In this case, the correct value is negative.

The complete equation for the tibia angle is shown in Appendix C for reference. Note that it includes the variable $?_4$ defined previously.

Tibia Mechanism Optimization

The equations of motion for the mechanism that drives the tibia are now calculated. This means the tibia angle can be expressed as a function of the input rotation and the mechanism link lengths as well as the constant angle between the two links extending off the input shaft. Once these equations are found, the `fmincon` function in MATLAB can be used to determine what link length values minimize the error between the desired motion and the motion that is actually produced by the mechanism. This error function is,

$$f = S\ [\ (\ (\ ?_f - \text{femurangle\_desired}\ )^2\ ) +\ (\ ?_t - \text{tibiaangle\_desired}\ )^2\ )^2\ ]$$

$?_f$ and $?_t$ are the angles of the femur and tibia, respectively, defined as functions of the lengths of r_1, r_2, r_3, r_4, r_5, r_6, r_7, a (the angle between r_2 and r_5), and $?_2$ (the rotation of the input link). The values of femurangle_desired and tibiaangle_desired are the desired motions. The lengths of the links in the four-bar mechanism driving the femur are kept at their optimized values found previously when computing the optimal lengths for the tibia mechanism links. A plot of the desired tibia motion and the tibia motion produced from the optimized linkage are superimposed in Figure 11 along with the previous results of the femur optimization. Notice the femur results are the same as in Figure 8. The femur parameters were held constant when optimizing the motion of the tibia. The femur motion is already optimized, and its motions should not be changed. This means the femur mechanism parameters are the same as before.

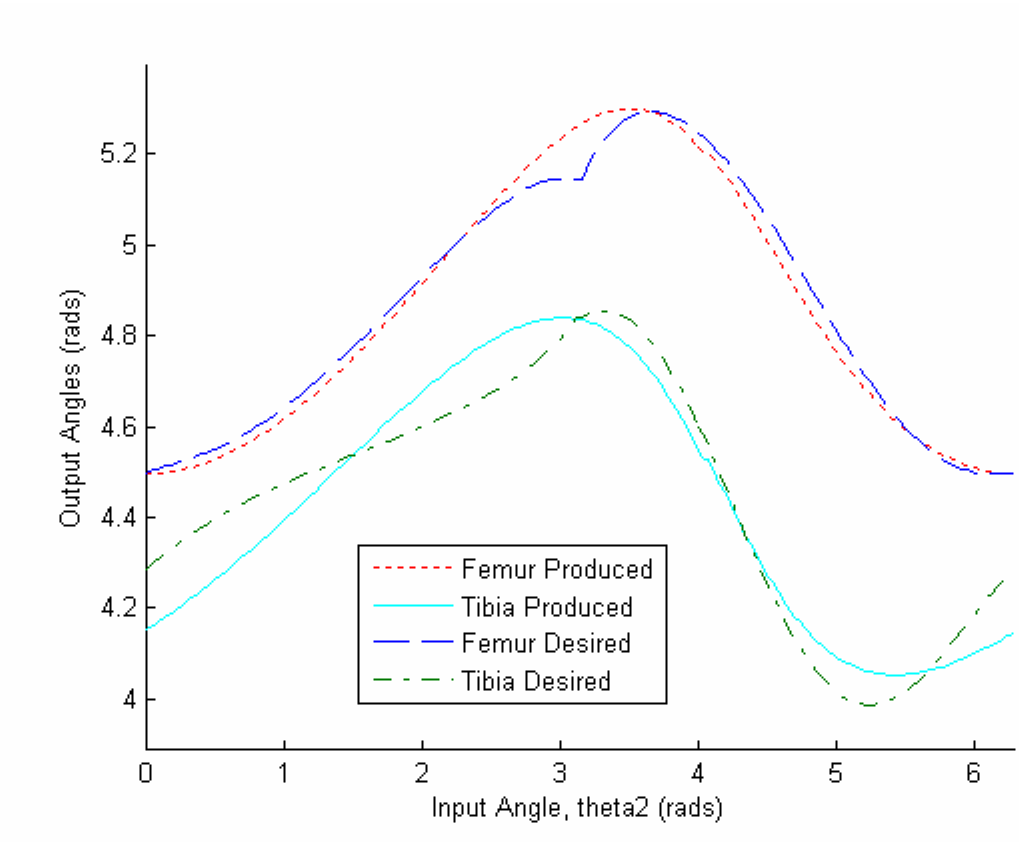

Figure 11 – Desired and Produced Tibia Motion

The parameters of the mechanism that create the motion plotted in Figure 11 are listed in Table 3. The links represented by the parameter name can be found in Figure 10. One assumption is that the femur and tibia lengths are equal. This can be seen in Table 3. Also, recall that $a = q_5 - q_2$.

Table 3 - Optimized Four-Bar Femur with Tibia Mechanism Parameters

Parameter	Value
r_1	1.000
r_2	0.269
r_3	1.353
r_4	0.925
r_5	0.703
r_6	2.276
r_7	2.450
r_f	3.053
r_t	3.053
a (rads)	5.631

Consideration for whether or not the optimized mechanism is physically feasible is also important. The optimized mechanism that creates the motion shown in Figure 11 was examined visually using MATLAB, and the path of the foot was also plotted to aid in analysis of the result. The mechanism is seen in Figure 12.

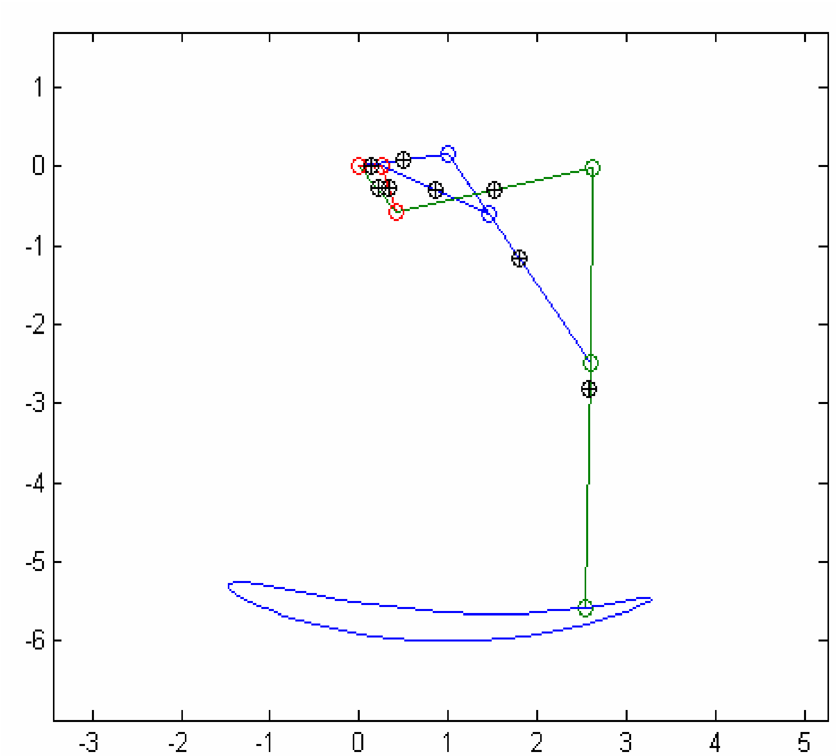

Figure 12 - Optimized Mechanism

From visual inspection of Figure 12, it is clear that this mechanism is physically feasible. None of the link lengths dwarf each other. The input link is noticeably smaller than the other links by roughly an order of magnitude, but this is not such an extreme difference that the mechanism would be unreasonable to build.

From examination of Figure 11, it is clear that the motion of the tibia that is produced by the optimized mechanism deviates noticeably from the desired motion. There is a limit on how close the actual motion can come to the desired motion, and that limit is partially due to the number of design variables available to optimize. In order to have more variables available to optimize, and therefore better produce the desired motion, more links must be added to the mechanism.

Femur Six-Bar Mechanism

In order to improve the motion of the femur about the point of foot touchdown and improve the motion of the tibia, two more links were added to the mechanism that controls the femur. Two links were added because in order to maintain a single degree of freedom, an even number of links must be added. The same process used to arrive at the original four-bar mechanism was used here to arrive at a six-bar mechanism. The first step in the process was to use Robert's Animator to get a general idea of what configuration the mechanism must take in order to produce the desired motions. After manipulation of the original four-bar mechanism, the six-bar mechanism shown in Figure 13 was established.

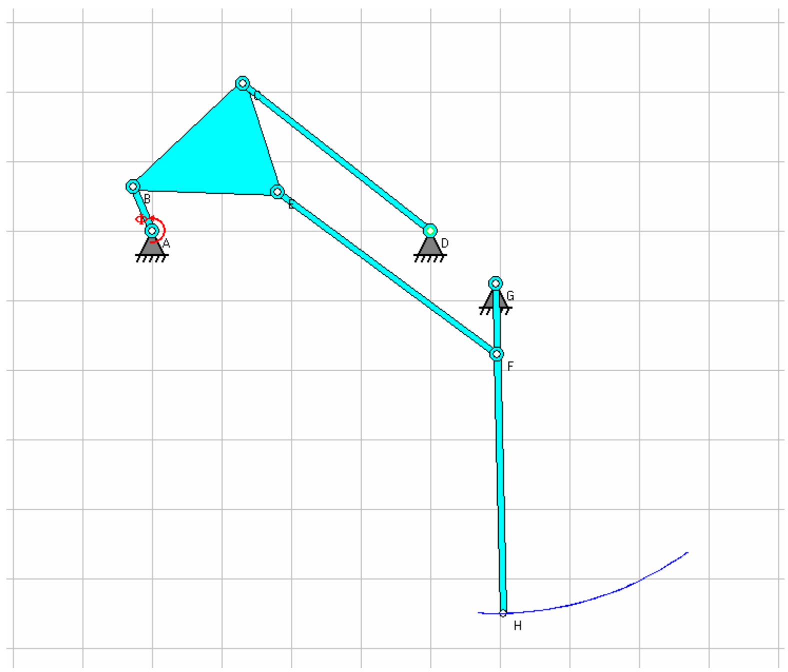

Figure 13 – Femur Six-Bar Linkage Conceptual Design

In Robert's Animator, the femur angle is plotted against the rotation of the input link. This gives a good idea of what type of mechanism is able to produce the hitch motion that is seen about the foot touchdown event. The plot of the femur angle as a function of the rotation of the input link as seen in Robert's Animator is shown in Figure 14.

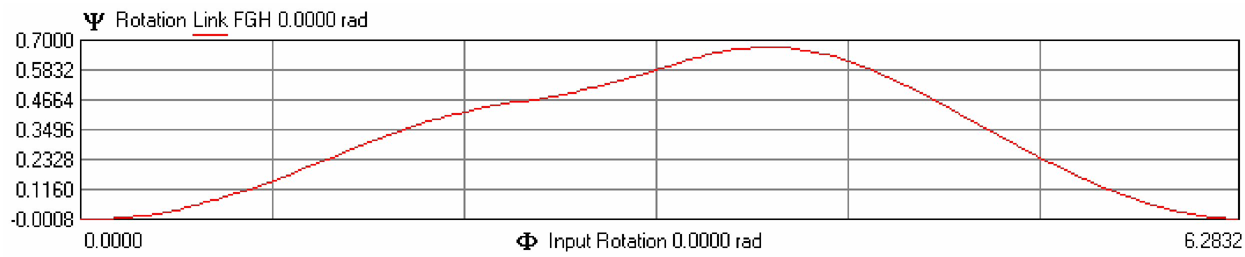

Figure 14 - Femur Motion Plotted in Robert's Animator

It is evident from Figure 14 that the mechanism has potential to more closely represent the foot touchdown event. In order to investigate just how close this six-bar linkage could match the desired motion of the femur, some more intense optimization of the mechanism was performed. This optimization was performed in the same manner as the optimization for the original four-bar femur mechanism and the complete six-bar mechanism that included the tibia. The first step is to develop the equations of motion for this mechanism.

Femur Six-Bar Equations of Motion

As discussed earlier, the first step in producing the equations of motion is to represent the mechanism as a set of vectors. The vectors chosen to represent this mechanism are shown in Figure 15.

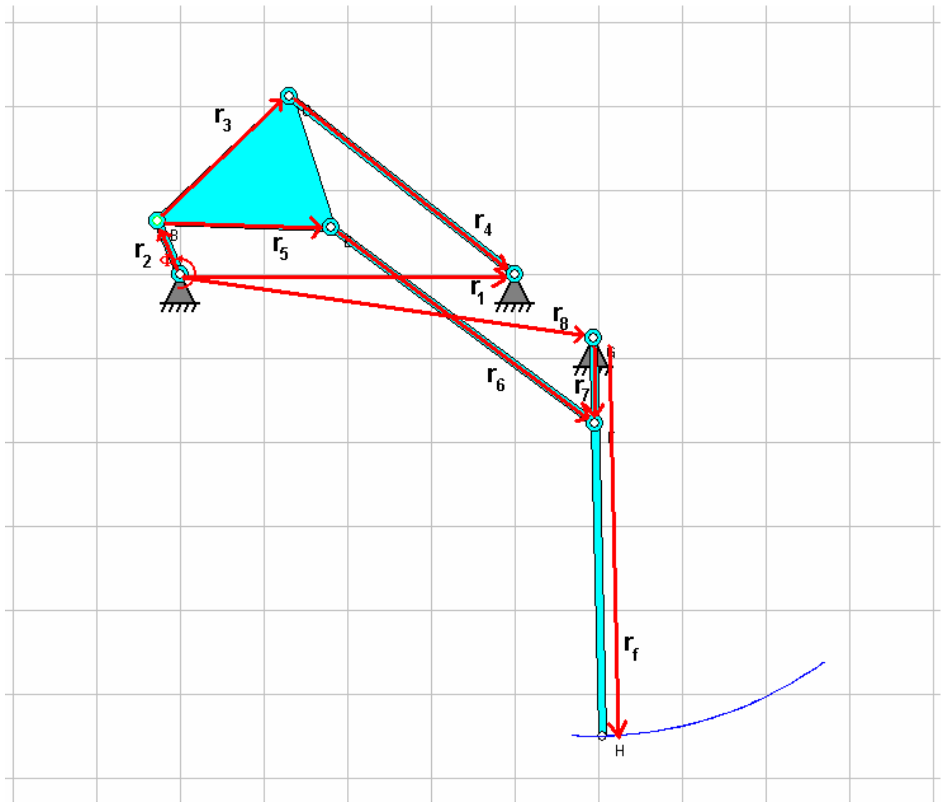

Figure 15 - Femur Six-Bar Mechanism Vectors

In order to produce the equations of motion for the femur, two vector loops must be defined. The first vector loop is the same as the vector loop for the four-bar femur mechanism.

$$\overline{r_1} + \overline{r_4} = \overline{r_2} + \overline{r_3}$$

The procedure outlined previously is followed in order to obtain an equation for q_4 as a function of q_2 and each link length in the vector loop. This equation is then used to find q_3 as a function of q_2 and each link length in the vector loop. The next step is to define a constant b as the angle between $\overline{r_3}$ and $\overline{r_5}$. Now, the second vector loop equation may be written as shown below.

$$\overline{r_6} = \overline{r_8} + \overline{r_7} - \overline{r_2} - \overline{r_5}$$

This equation is expressed in its x and y components as before, each component equation is squared, and they are added together. Then, the equation $A\cos(q_7) + B\sin(q_7) + C = 0$ is defined, and t is expressed as $t = \dfrac{-B + s\sqrt{B^2 - C^2 + A^2}}{C - A}$. It is noted that the coefficients A, B, and C have different values than for the previous two cases. The A, B, and C coefficients must be determined with regards to the six-bar femur mechanism vectors.

Where, as before, $s = \pm 1$. Finally, q_7 is found from the equation $q_7 = 2\tan^{-1}(t)$. The angle of the femur is then known because it is equal to the angle of link 7. The femur is now a function of the input link rotation, the length of each link in the mechanism, and the one solid link angle b. From inspection of the produced mechanism, it was found that the proper value of s is -1 for the desired mechanism configuration. The equation for the angle of the femur is in Appendix D.

Femur Six-Bar Mechanism Optimization

With the equations of motions for the potentially improved six-bar femur mechanism complete, the same process as described before using the `fmincon` function may be used in order to optimize the motion. The motion that is produced from the optimized femur mechanism is shown in the figure below.

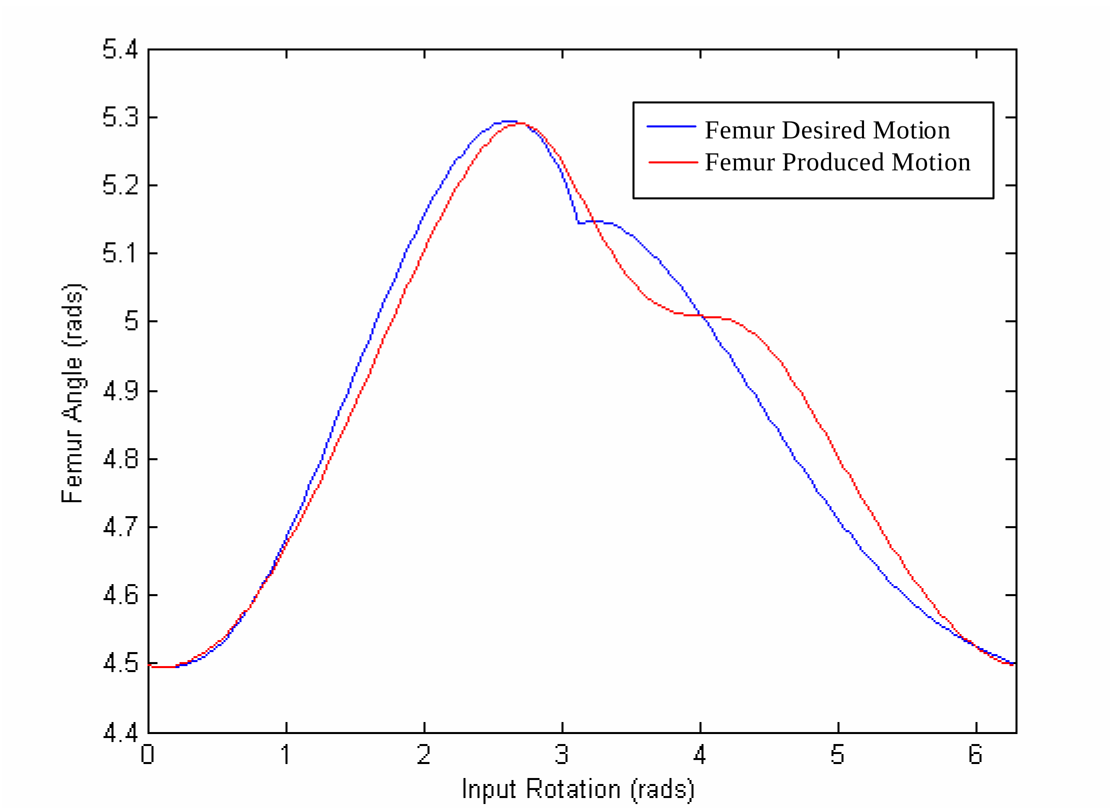

Figure 16 - Six-Bar Femur Optimization Result

The optimized parameters that produce the motion shown in Figure 16 are listed in Table 4 below. The links represented by each parameter can be found in Figure 15. Also, recall that $b = q_3 - q_5$.

Table 4 - Optimized Six-Bar Femur Mechanism Parameters

Parameter	Value
r_1	1.000
r_2	0.175
r_3	0.532
r_4	0.863
r_5	0.572
r_6	0.922
r_7	0.274
r_8	1.223
q_8 (rads)	-0.139
b (rads)	0.890

It is evident from Figure 16 that the optimized femur motion produced by the six-bar linkage does not perfectly match the desired motion. It was originally hypothesized that a mechanism with more links should be better able to match the desired motions because there are more design variables. The results obtained from this research do not validate this hypothesis. It is possible that the four-bar linkage is simply a better mechanism configuration for matching the desired femur motion. The range of motions that the six-bar linkage can create are constrained and will not produce motions better than the four-bar linkage. It is also reasonable to assume that the implementation of the `fmincon` function was not optimal and did not produce the best possible results. Further investigation of this phenomenon would need to be undertaken in order to arrive at a conclusive result. However, the motions that are produced are still reasonable. It is worth investigation how well the desired tibia motions can be matched when the femur is being controlled by the six-bar linkage. In the next section, this will be discussed.

Tibia Eight-Bar Mechanism Consideration

The tibia mechanism chosen to work in conjunction with the six-bar femur mechanism is similar to the tibia mechanism that was chosen to be used with the four-bar femur mechanism.

Again, Robert's Animator was utilized in order to analyze a preliminary mechanism configuration for the tibia. Figure 17 shows the mechanism.

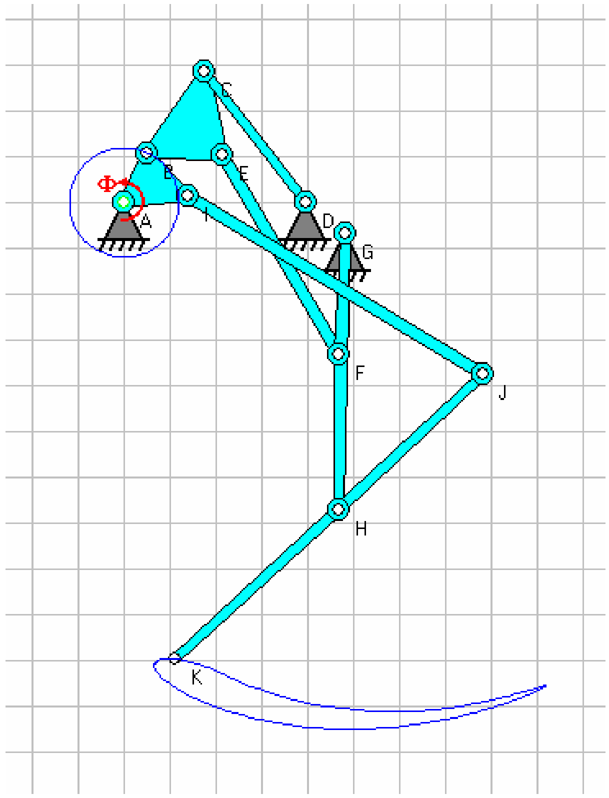

Figure 17 – Tibia with Femur Six-Bar Linkage Conceptual Design

The mechanism shown here consists of eight links, one of which is the ground link that represents the body of the robot. This is the link between points A, D, and G, as seen in Figure 17.

Tibia Eight-Bar Equations of Motion

The procedure for producing the equations of motion of the mechanism in Figure 17 is now carried out. The vectors used to describe this mechanism are shown in Figure 18. For the sake of clarity, only the new vectors associated with the tibia mechanism are shown in the figure. The vectors from Figure 15 still apply here.

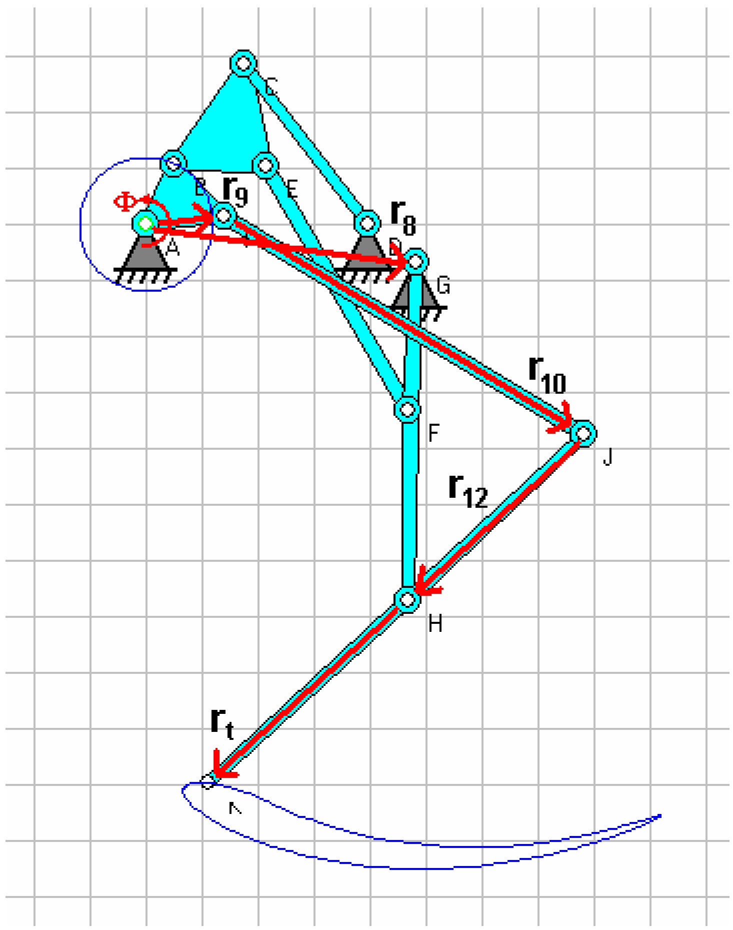

Figure 18 - Tibia Mechanism Vectors with Six-Bar Femur

Incorporating the vectors defined for the femur mechanism and the vectors shown in Figure 18, the equations of motion for the mechanism can be produced. The tibia angle can be described as a function of the length of every link in the mechanism, three solid link angles, and the rotation of the input link. The three solid link angles are one on the input link, the second on the triangle link in the femur mechanism, and the third is the angle of vector r_8 that makes up one side of the base link which is the body of the robot. The equation for the tibia angle is shown in Appendix D.

Tibia Eight-Bar Mechanism Optimization

Using the same procedure as before to optimize this mechanism, the motions achieved are shown in Figure 19.

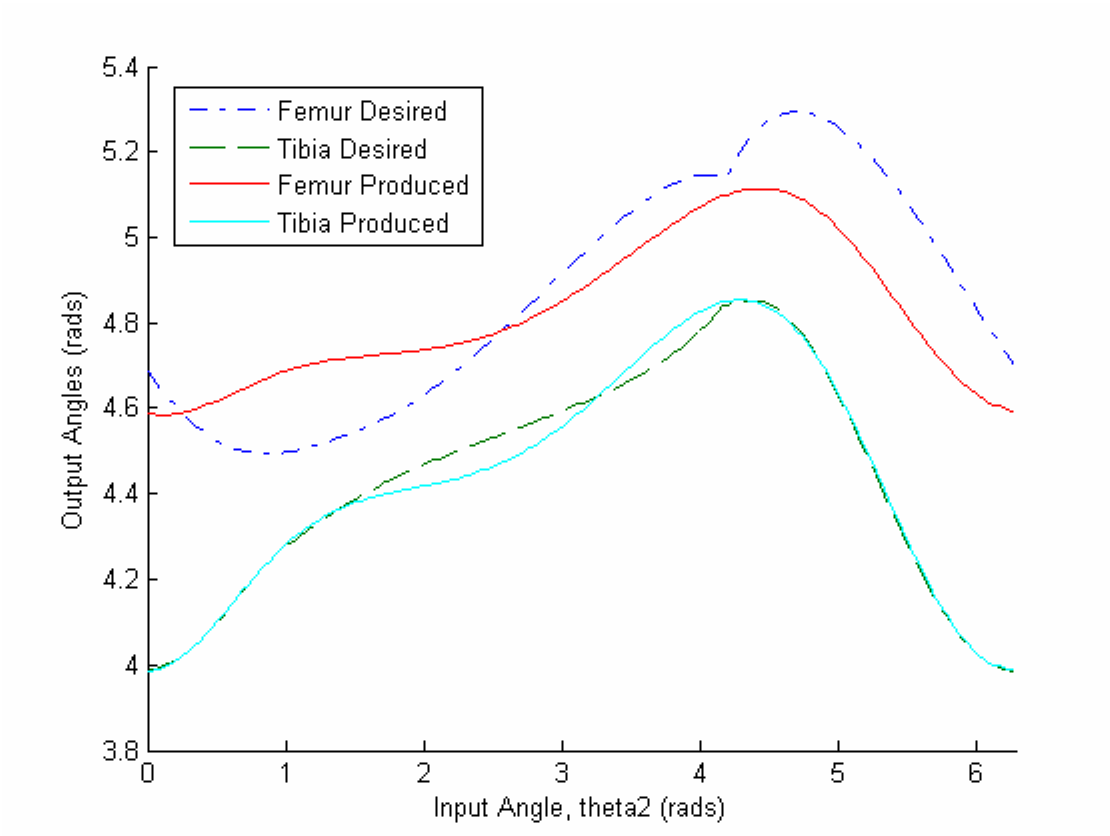

Figure 19 – Desired and Produced Tibia and Femur Motions

The parameters of the mechanism that create the motion shown in Figure 19 are listed in Table 5. The links represented by each parameter can be found in Figure 18.

Table 5 - Optimized Six-Bar Femur and Tibia Mechanism Parameters

Parameter	Value
r1	1.000
r2	0.235
r3	0.529
r4	0.892
r5	0.542
r6	1.183
r7	0.646
r8	1.208
r9	0.168
r10	2.660
r11	2.621
r12	1.131
r13	2.660
beta	0.927
theta8	-0.153
alpha	-0.194

Recall that a is the angle between the r_2 vector and the r_9 vector, $a = q_2 - q_9$. In the actual optimization code, the r_6 and r_{11} values were not optimized. The reason for this is because the code would not run if these values were included in the optimization. Instead, specific values were picked for these two vectors that were found in the Robert's Animator software. More discussion with regard to dealing with these types of issues when using the `fmincon` command can be found in Chapter 3.

It is clear that there is a considerable amount of error for the femur motion. As an attempt to alleviate this problem, the mechanism parameters that affect the femur motions can be made equal to their optimized values found in the optimization of only the femur angle. However, when the femur mechanism parameters are constrained to these values, the optimization routine cannot find a real solution for the tibia parameters. This is to say, there is no configuration of the mechanism in which the femur motions are closely matched and the tibia motions are also closely matched. This is why in Figure 19, the tibia motions are close to the desired motion and the femur motions are significantly different than the desired motions. The limits of the capabilities of the mechanism have been reached.

Alternative Optimization – Foot Path

Until now, the optimization used in this procedure has intended to decrease the error between the desired femur and tibia angles and the actual tibia and femur angles. Another method for optimizing the mechanism in order to produce the desired motion is to optimize the path of the foot point. The equations of motion for the mechanism have already been produced so the equation describing the location of the foot point in x and y coordinates is easily determined. The assumption that the femur and tibia are of equal lengths is assumed for this optimization, as in all of the other optimizations. Again, the `fmincon` function is used in MATLAB to find the link lengths that minimize the error between the desired foot path and the foot path produced by the mechanism. The error function was defined as,

$$f = S\ [\ (\text{footpath\_actual}_x - \text{footpath\_desired}_x)^2 + (\text{footpath\_actual}_y - \text{footpath\_desired}_y)^2\ ];$$

foot_x_actual and foot_y_actual are the x and y coordinates of the footpath of the mechanism. These values are functions of the mechanism parameters. As with the angle optimization, 200 points were chosen to represent the desired and actual footpath locations.

Below is a figure of the desired foot path and the optimized mechanism footpath. This is for the mechanism that includes a six-bar femur portion, the more complicated of the two mechanisms considered.

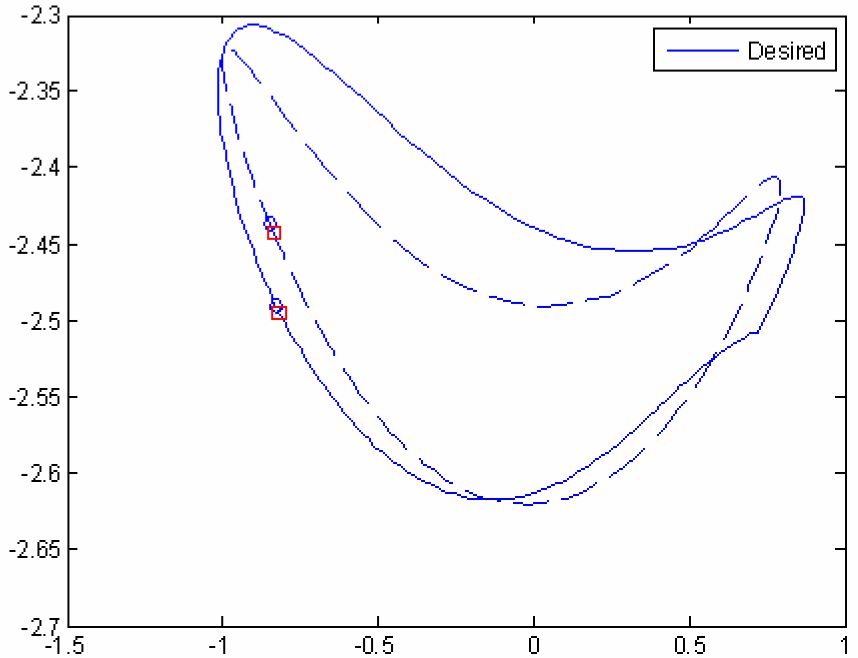

Figure 20 - Footpath Optimization

The mechanism parameters that create the motions in Figure 20 are listed in Table 6.

Table 6 - Mechanism Parameters from Footpath Optimization

Parameter	Value
r1	1.000
r2	0.155
r3	0.645
r4	0.800
r5	0.570
r6	1.200
r7	0.580
r8	1.115
r9	0.244
r10	1.708
r11	1.320
r12	1.100
r13	1.320
beta	0.810
theta8	-0.384
alpha	-0.255

Figure 20 shows that there are noticeable errors in the actual motion when compared to the desired motion. The circle and square points on each foot path are used to visually ensure that the produced foot path is tracing in the same direction as the desired footpath. The circle point is clockwise from square point in each case, which ensures the direction of foot travel is correct.

Correlation of Foot Path Optimization and Leg Angle Optimization

Two methods for optimizing the mechanism have just been discussed. The first method is optimizing the mechanism parameters to minimize the error between the desired femur and tibia angles and the actual femur and tibia angles. The next method was to optimize the mechanism parameters in order to minimize the error between the desired footpath and the actual footpath of the mechanism. It is worthwhile to analyze the effect that each method of optimization has on the mechanism.

When the femur and tibia angle optimization method is performed, a set of mechanism parameters is obtained. These parameters define the equations of motion for the mechanism. Using these parameters, it is possible to plot the path of the foot produced. Even though the foot path is not directly being optimized, it is still desired that the actual foot path matches the desired foot path. The result of computing the actual footpath when optimizing for the femur and tibia angles is shown in Figure 21.

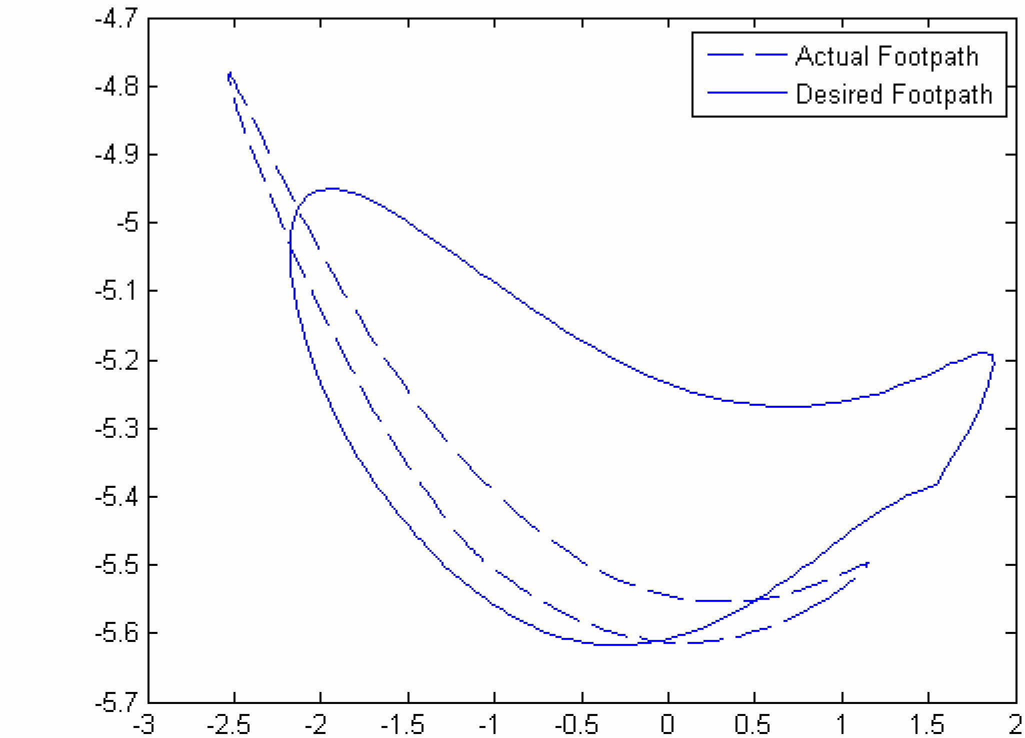

Figure 21 - Footpath Produced with Parameters from Femur and Tibia Angle Optimization

In Figure 21, a noticeable difference between the actual and desired footpath is observed. It is concluded that optimizing the mechanism to produce desired femur and tibia angles does not ensure a suitable footpath will be developed.

Similar to plotting the footpath motions with parameters optimized to match the desired femur and tibia angles, one could plot the femur and tibia angles with parameters optimized to match the desired footpath. The result of computing the actual femur and tibia angles when using parameters optimized for the desired footpath is shown in Figure 22.

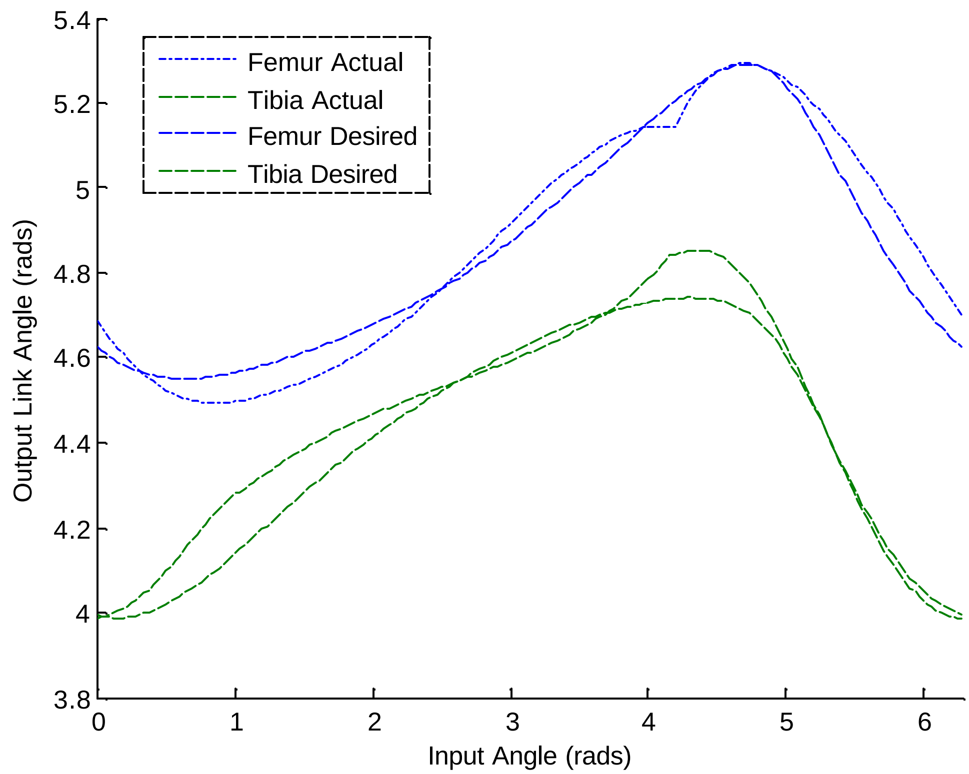

Figure 22 - Femur and Tibia Motions when Optimizing Footpath

The results obtained from the footpath optimization are better than those obtained through the femur and tibia angle optimization in Figure 19. This result is interesting because the angle of the femur and tibia are not the values that are directly being optimized. The reason why a better match to the desired femur and tibia angles is obtained using the footpath optimization is due to the fact that there is indeed a correlation between the footpath and the angle of the femur and tibia. The error function used for the footpath optimization is better suited for finding parameters that match the desired motion. The mechanism parameters obtained from the footpath optimization can now be used in the angle optimization method as initial guesses. The result of this is shown in Figure 23.

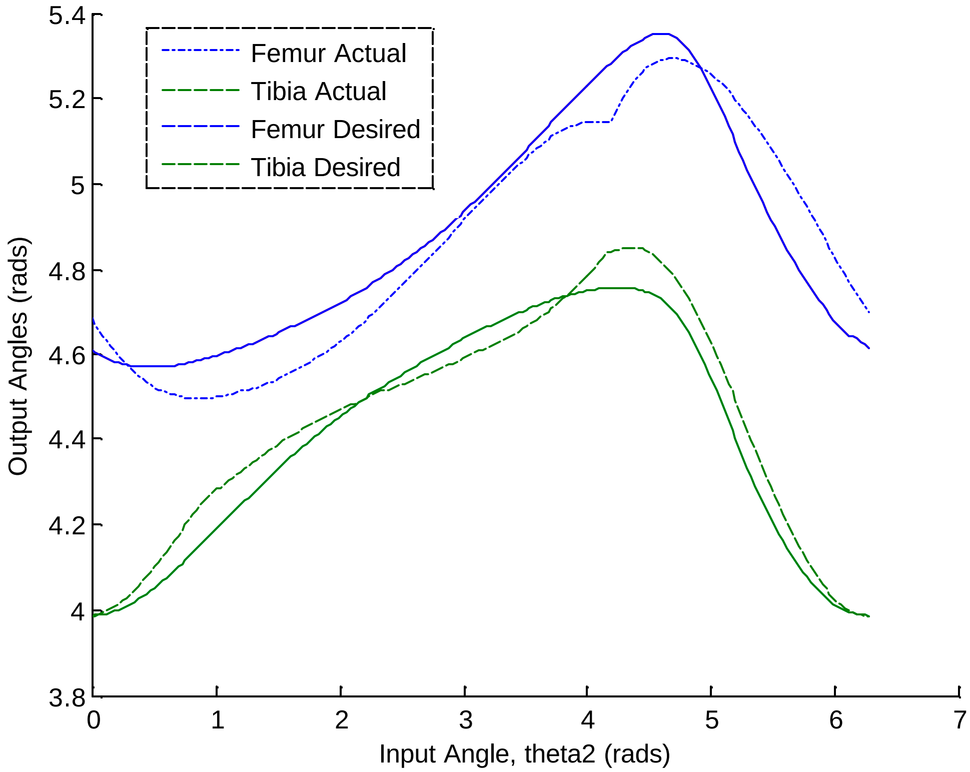

Figure 23 - Angle Optimization Using Values Obtained from Footpath Optimization

The resulting plot is very similar to that in Figure 22. The optimization routine does not improve the motions any significant amount.

3. Results and Discussion

The result of this research is a systematic approach to designing a single-degree-of-freedom mechanism, consisting of only revolute joints, which generates motions that represent a stable walking gait. This systematic approach was carried out, and a mechanism was designed. The optimization is an integral part of this process and utilizes the `fmincon` function in MATLAB. This function will be discussed. The mechanism that was designed reasonably matches a set of desired motions that are known to provide stable walking. The values for each parameter of the final mechanism are tabulated.

Design Process

The basic steps to the design process are to define the desired motions. Next, a mechanism must be created that is reasonably capable of matching the desired motions. A mechanism design software package such as Robert's Animator can aid in this step, however, it is not capable of performing the optimization necessary in the following steps. After the mechanism has been created, the equations of motion for the mechanism must be defined. These equations of motion must define the angle of the leg links (the femur and tibia) as a function of the input link, the link lengths of the entire mechanism, and any solid link angles. With the desired motions known and the equations of motions developed, it was found that the `fmincon` function in MATLAB is a very useful tool for optimizing the mechanism.

MATLAB `fmincon` Function

When implementing the `fmincon` function, a number of best practices were determined. These best practices help in avoiding errors, pinpointing sensitive parameters of the optimization, and allowing MATLAB routines to run more smoothly.

It was found that starting with the optimization of a single parameter is helpful. This method was used for the purposes of this research. In the four-bar linkage femur mechanism optimization, the input link r_2 was the sole parameter at first. There is no reason why this was the first parameter used in the optimization. The fact that it is link 2, or that it is the shortest link, or that it is the input link, had no bearing on why it was used first. The parameter r_2 was simply the first undefined parameter from the equations of motion. Any of the other parameters could have been used first with equal results.

It is also suggested to tightly constrain this parameter with upper and lower bounds very close to the initial guess. When parameters are initially added to the `fmincon` function, their upper and lower bounds were typically constrained to within ±1% of the initial guess. Constraining a single parameter at first is helpful because it is a good method for determining whether or not the code is written correctly. Once it has been verified that the code is working correctly, the bounds on the first parameter may be expanded, and more parameters may be added. It is helpful to add a single parameter at a time to the code.

It was also determined that an iterative approach of running the optimization code can be helpful. This means that the code is run a first time with a primary initial guess. A result for the optimized value of that parameter is returned. The optimized value of the parameter can then be used as the initial guess of a second optimization. The upper and lower bounds of the parameter can be adjusted to more evenly straddle the new initial guess value. This method can be done one parameter at a time, or several parameters may be adjusted with each optimization. If parameters are sensitive, it can be helpful to run fewer parameters in a single optimization until adequate initial guesses for all the parameters are found.

When optimizing the complete walking mechanism that included a six-bar linkage for the femur and another four-bar linkage for the tibia, two vectors could not be optimized. This is to say that when these two parameters were used in the optimization code with the `fmincon` function, the code would lock up and not produce a result. The reason for this is believed to be that these are very sensitive values. When the `fmincon` routine tests a value a slight distance away from the initial guess, the mechanism fails and does not produce a result. This means that the input link cannot make a full rotation. The two values that could not be optimized were the r_6 and r_{11} vectors, representing the link lengths of links 6 and 11, respectively. It was only determined that these two links posed a problem when one link parameter at a time was successfully added to the optimization routine. As each new parameter was added, the upper and lower limits were constrained tightly to the initial guess (obtained from Robert's Animator). Once the code ran successfully, the upper and lower bounds were increased.

Higher Linkage Mechanisms

It is reasonable to assume that a mechanism with more links would have more design freedom in that it has more adjustable parameters (link lengths). However, it was found that increasing the femur mechanism from a four-bar linkage to a six-bar linkage did not produce a significantly better result for the femur motion. To illustrate this point, the error associated with the four-bar linkage femur mechanism and the error associated with the six-bar linkage femur mechanism are shown here. Recall that the error value equals the summation of the square of the difference between the desired angle of the link and the actual angle of the link. The desired motion and actual motion are vectors of length 200, so the error value is the summation of 200 values.

Table 7 – Four-Bar and Six-Bar Femur Mechanism Error

Four-Bar Femur Mechanism Error	Six-Bar Femur Mechanism Error	Percent Diff (%)
0.292	0.574	0.53

The errors found when using a six-bar mechanism were found to be considerably larger than those found when using a four-bar mechanism. This is somewhat counter intuitive because the six-bar linkage has more links and therefore more design freedom. There are two possible explanations for this phenomenon. The four-bar linkage could simply be a much better choice of a mechanism for matching the desired motion. It is simply a constraint of the mechanism configuration that does not allow the six-bar mechanism to match as well as the four-bar mechanism. The larger error for the six-bar femur mechanism could also be due to less than optimal implementation of the `fmincon` function. Further investigation would have to be carried out in order to realize the true reason.

When the tibia portions of each of these two mechanisms were added, it was a different outcome regarding which one produced a smaller error value. The error associated with the best optimized mechanism with a four-bar femur portion, and the error associated with the best optimized mechanism with a six-bar femur portion are shown below in Table 8.

Table 8 - Four-Bar Femur plus Tibia and Six-Bar Femur plus Tibia Error

Four-Bar Femur Mechanism Error	Six-Bar Femur Mechanism Error	Percent Diff (%)
2.133	1.447	0.32

It is clear from Table 8 that the mechanism with more links produces better motions. This is in accordance with the initial assumption that adding more links would decrease the error.

Final Mechanism

The mechanism that has been determined to be most favorable is the mechanism with the four-bar femur portion. It is noted that this mechanism does not produce the smallest error from the desired motions. The mechanism with a six-bar femur portion produced the smallest error. The reason the mechanism with a six-bar femur portion was not chosen was because of its complexity. It was determined that the benefits of the better motions are outweighed by the complexity of its design. The optimal mechanism with the four-bar femur portion is represented in Figure 24. The vectors representing the links are the same as those described in Figure 10. This figure shows the mechanism with each link length set to its optimized value.

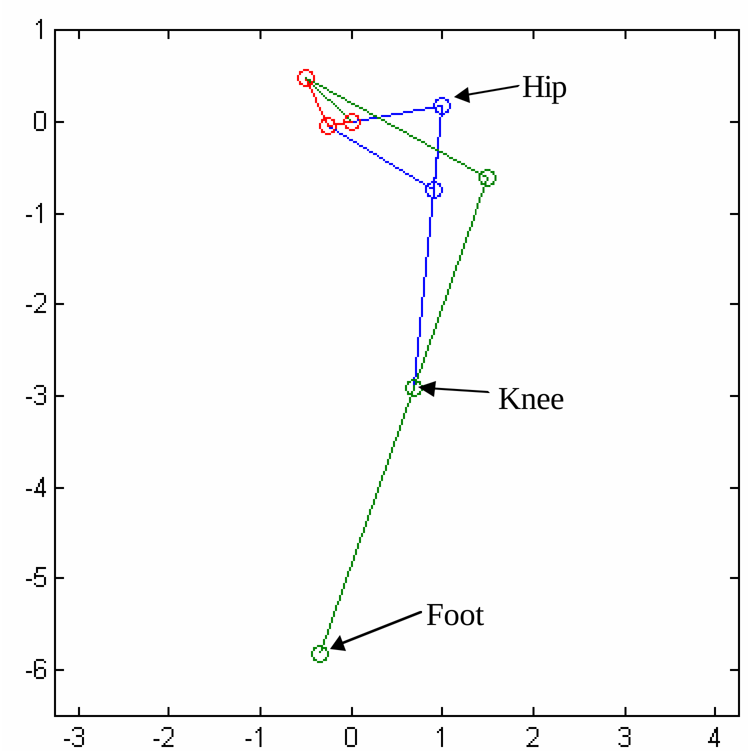

Figure 24 - Final Optimized Mechanism

The mechanism parameters corresponding to the final optimized mechanism with a four-bar femur portion are listed here in Table 9. These are the same values listed in Table 5; however, they are listed again here in the results because they represent a significant part of the final results of this research.

Table 9 - Final Optimized Four-Bar Femur and Tibia Mechanism Parameters

Parameter	Value
r_1	1.000
r_2	0.264
r_3	1.358
r_4	0.916
r_5	0.703
r_6	2.276
r_7	2.450
r_f	3.053
r_t	3.053
alpha (rads)	5.631

Recall that a is the angle between q_5 and q_2, $a = q_5 - q_2$. It is also noted that a mechanism is completely scalable. This means that if each link length is multiplied by a scaling constant, the mechanism will still produce the same motions.

The design procedure and mechanism described above meets the objective set forth for this research of designing a single-degree-of-freedom mechanism that reasonably produces motions similar to those of a stable biped walking gait.

4. Future Work

This research begins to explore the method of creating biped walking motions using a single-degree-of-freedom mechanism. There is a large amount of future work that may be extrapolated from this research.

Walking Simulation of Current Model

The mechanism designed in this research was done so with regard only to its ability to match a desired set of motions known to produce stable walking. Even if a perfect match to these motions could be achieved, it would still not be a guarantee that this mechanism could actually walk and not trip or fall over. The dynamics associated with the many links that make up the mechanism may alter the robots stability. With multiple links rotating, the dynamics of this mechanism will be different than those of a robot with only femur and tibia links controlled by motors.

In this research, the mechanism does not match the desired motions exactly, and there is in fact a significant amount of error. In order to achieve a stable walking mechanism, it may be necessary to include parameters relating to the stability of the mechanism into the optimization of the link lengths. One such method for analyzing the stability of a mechanism is a model created by Dr. E. Westervelt called SHS Sim. Using this model, it is possible to simulate the walking of this mechanism and inspect its stability. Preliminary use of this model has already begun; however, no results have been obtained at this point in time.

Analysis of Other Mechanism Configurations

Further investigation into the best mechanism may still be performed. The addition of links to the current mechanism would allow for more design variables and a greater range of possible motions. With more links, the mechanism becomes increasingly complicated; however, the benefits of a closer match to the desired motions may outweigh the negatives of a complicated mechanism. The process outlined in this research may be used for a mechanism with any number of links.

Furthermore, this research only considers the use of revolute joints in the mechanism. It is reasonable to presume the use of prismatic (slider) joints could lead to a new mechanism that can represent biped walking motions. If prismatic joints were used, the equation of motion generation for the mechanism would be slightly different. With revolute joints, the vectors used to create a vector loop equation are all of fixed length because the links are all of fixed length. With a prismatic joint, there will be at least one vector that is not of a fixed length. A full explanation of how to handle prismatic joints when representing a mechanism with vectors is found in a kinematic mechanism design reference book [8].

There is one final modification that could be made to the mechanism developed in this research. The link that consists of the femur and r_7 does not have to be a straight link. This link could be shaped more like a triangle with the three corners consisting of the foot, the knee, and the joint that connects r_6 to r_7. This would present one more variable that may be optimized and could make the motion of the mechanism closer to that of the desired motions.

Force Transmission

Another important aspect of the mechanism that has not been considered is force transmission. A thorough study of the transmission of forces through each link and joint in the mechanism should be undertaken before a mechanism could be considered reasonable to build. The force transmission through the mechanism could be included in the optimization procedure in order to produce a mechanism that not only tries to match the desired motions, but does so in a way that minimizes the forces transmitted through the mechanism. One particular area of concern regarding force transmission is the fact that the input link is rather small compared to the rest of the links. The length of the link represents the moment arm for transmitting a force, so in order to have a large force, because the moment arm is so small, a large torque is needed. The physical limitations of materials and other parts such as joints and motors should be analyzed before attempting to build a working model of this mechanism.

Physical Model

A physical model of a mechanism designed using the procedure outlined in this thesis could be created. Building the physical model would involve choosing appropriate materials for the links, appropriate revolute joints, and an appropriate motor. The motor could only be chosen after the previously mentioned force transmission analysis was performed. In order to implement control strategies, it would also be necessary to have a sensor of some type. This sensor would likely be an encoder that would measure the angle of a moving link. In building the physical model, it may also be wise to make the link lengths adjustable. Adjustable link lengths in the physical model could facilitate minor link length changes that could dramatically affect the walking motion. This could be useful for walking at different speeds or on different surface slopes, or simply for trial and error of finding the best link lengths. This is validated by the fact that during optimization, small changes in the parameters being optimized often produced dramatic differences.

Other Applications

Finally, it should be noted that this design procedure could apply to any application in which there is a defined desired motion for a rigid body. The process used for designing the walking mechanism in this research could easily be applied to any field outside bipedal locomotion.

5. Conclusions

Bipedal locomotion is of unique interest due to its implications for human pathology in terms of rehabilitation and prosthetic design. Current biped robots require multiple actuators (motors) because they have multiple degrees of freedom. Furthermore, they require complex control strategies to enable stable walking. These two characteristics have resulted in biped prototypes that are complicated, expensive, heavy, and energetically inefficient.

In order to alleviate many of these deficiencies, it was stated that a single-degree-of-freedom mechanism could provide stable walking motions. A systematic approach for designing this single-degree-of-freedom kinematic mechanism that produces biped walking motions has been developed.

The first step is to define the desired motions for the femur and tibia. The next step is to find a general linkage configuration that is believed to reasonably represent the desired motion. For the purposes of this research, Robert's Animator mechanism design software is used to analyze preliminary mechanism designs. After the preliminary mechanism configuration is determined, the equations of motion for the mechanism must be determined. The angles of the femur link and tibia link should be determined as functions of the dimensions of each link in the mechanism, as well as the input link rotation. Once the equations of motion for the mechanism have been determined, they can be combined with the desired leg motions in a MATLAB script similar to that found in the Appendix B using the `fmincon` optimization function. The `fmincon` function will find and display the link length values that minimize the error between the desired leg motion and the motion produced by the mechanism. Visual inspection of the resulting mechanism is then performed to ensure the results are sufficient. The visual inspection of the mechanism may be performed using MATLAB, a physical model, or mechanism design software such as Robert's Animator.

References

[1] Berns, K. *Walking Machine Catalogue*. http://www.walking-machines.org, 2005.

[2] American Honda Motor Co. Inc., ASIMO: The World's Most Advanced Humanoid Robot. http://asimo.honda.com, 2005.

[3] McGeer, T. (1990) Passive dynamic walking, *International Journal of Robotics Research*, Vol. 9, No., 2, pp. 62-82.

[4] Lilly, B. *Design and Analysis of a Mechanically Coordinated, Dynamically Stable, Quadruped Trotting Machine.* M.S. Thesis, The Ohio State University, 1986.

[5] E. R. Westervelt, G. Buche, and J. W. Grizzle, "Experimental Validation of a Framework for the Design of Controllers that Induce Stable Walking in Planar Bipeds," Preprint, The International Journal of Robotics Research, Vol. 24, No. 6, June 2004, pp. 559-582.

[6] E. R. Westervelt, J. W. Grizzle, and D. E. Koditschek, "Hybrid Zero Dynamics of Planar Biped Walkers," *IEEE Transactions on Automatic Control*, Vol. 48, No. 1, January 2003, pp. 42-56.

[7] Raibert, M.H. *Legged Robots That Balance*. MIT Press, Mass., 1986
Walkers. IEEE Transactions on Automatic Control, Vol. 48, No. 1, January 2003.

[8] Kinzel, G.L., and Waldron, K.J. *Kinematics, Dynamics, and Design of Machinery* 2nd Ed., Wiley, 1999.

[9] A. Dunki-Jacobs, BIRT Biped Robot, Graduate Research, The Ohio State University, Department of Mechanical Engineering.

[10] Heron Technologies, Netherlands, 2006. http://www.heron-technologies.com/roberts/

Appendix

Appendix A: Four-Bar Femur Linkage With Tibia Equation of Motion Generation Using Maple

```
> restart;
> #4 Bar Femur Function Generation
> r3c3 := r1*cos(theta1) + r4*cos(theta4) - r2*cos(theta2):
> r3s3 := r1*sin(theta1) + r4*sin(theta4) - r2*sin(theta2):
> r3squared := simplify(r3c3^2 + r3s3^3):
> A := 2*r1*r4*cos(theta1) - 2*r2*r4*cos(theta2):
> B := 2*r1*r4*sin(theta1) - 2*r2*r4*sin(theta2):
> C := r1^2 + r2^2 + r4^2 - r3^2 -
2*r1*r2*cos(theta1)*cos(theta2) -
2*r1*r2*sin(theta1)*sin(theta2):
> t:= (-B + sqrt(B^2 - C^2 + A^2)) / (C - A):
> theta4 := simplify(2*arctan(t));
> #6 Bar Tibia Function Generation
> r6c6 := r1*cos(theta1) + rf*cos(thetaf) - r5*cos(theta5) -
r7*cos(theta7):
> r6s6 := r1*sin(theta1) + rf*sin(thetaf) - r5*sin(theta5) -
r7*sin(theta7):
> r6squared := simplify(r6c6^2 + r6s6^2):
> AA := -2*r1*r7*cos(theta1) - 2*rf*r7*cos(thetaf) +
2*r5*r7*cos(theta5):
> BB := -2*r1*r7*sin(theta1) - 2*rf*r7*sin(thetaf) +
2*r5*r7*sin(theta5):
> CC := r1^2 + r5^2 + r7^2 - r6^2 + rf^2 +
2*r1*rf*cos(theta1)*cos(thetaf) -
2*r1*r5*cos(theta1)*cos(theta5) -
2*rf*r5*cos(thetaf)*cos(theta5) +
2*r1*rf*sin(theta1)*sin(thetaf) -
2*r1*r5*sin(theta1)*sin(theta5) -
2*rf*r5*sin(thetaf)*sin(theta5):
> tt := (-BB - sqrt(BB^2 - CC^2 + AA^2)) / (CC - AA):
> theta7 := simplify(2*arctan(tt));
```

Appendix B: Four-Bar Femur Linkage and Tibia Optimization Using MATLAB `fmincon`

```
function r_opt = opt_angle

clear, clc, close all

%initial guesses
r2_initial = .265;
r3_initial = 1.355;
r4_initial = .915;
rf_initial = 3;

r5_initial = .5;
r6_initial = 2.5;
r7_initial = 2;
alpha_initial = 1.5*pi;

initial = [r2_initial r3_initial r4_initial rf_initial r5_initial r6_initial
r7_initial alpha_initial] ;

%upper and lower bounds of each parameter
lb(1) = .26;    ub(1) = .27;
lb(2) = 1.35;    ub(2) = 1.36;
lb(3) = .91;    ub(3) = .92;
lb(4) = 2.9;    ub(4) = 3.1;

lb(5) = .2;     ub(5) = 1;
lb(6) = 2.1;    ub(6) = 3.3;
lb(7) = 1.8;     ub(7) = 2.5;
lb(8) = 1.49*pi;  ub(8) = 1.9*pi;

options = optimset('DiffMaxChange',1e-3,...
    'DiffMinChange',1e-4,...
    'Display','iter',...
    'MaxFunEvals',60,...
    'MaxIter',50,...
    'TolFun',1e-4,...
    'TolCon',1e-4,...
    'TolX',1e-5);

disp('fmincon')
[r_opt,fval] = fmincon(@err,initial,[],[],[],[],lb,ub,[],options);

femurangle_desired = [4.4974    4.5005    4.5034    4.5063    4.5093
4.5122    4.5152    4.5182    4.5213    4.5245    4.5279    4.5313    4.5349
4.5386    4.5424    4.5464    4.5506    4.5550    4.5595    4.5642    4.5691
4.5742    4.5795    4.5850    4.5907    4.5965    4.6026    4.6089    4.6153
4.6219    4.6288    4.6358    4.6430    4.6503    4.6578    4.6655    4.6734
4.6814    4.6896    4.6979    4.7064    4.7150    4.7237    4.7326    4.7415
4.7506    4.7598    4.7691    4.7785    4.7879    4.7974    4.8070    4.8167
4.8264    4.8362    4.8460    4.8558    4.8657    4.8755    4.8854    4.8953
```

```
4.9051    4.9150    4.9248    4.9345    4.9443    4.9539    4.9635    4.9730
4.9825    4.9918    5.0010    5.0101    5.0190    5.0278    5.0364    5.0449
5.0531    5.0611    5.0690    5.0765    5.0838    5.0909    5.0976    5.1040
5.1101    5.1157    5.1211    5.1260    5.1304    5.1344    5.1379    5.1409
5.1433    5.1451    5.1463    5.1468    5.1467    5.1458    5.1441    5.1441
5.1645    5.1831    5.2000    5.2152    5.2289    5.2410    5.2517    5.2610
5.2691    5.2759    5.2815    5.2860    5.2893    5.2917    5.2930    5.2935
5.2929    5.2916    5.2893    5.2863    5.2825    5.2779    5.2727    5.2667
5.2601    5.2529    5.2450    5.2366    5.2276    5.2180    5.2080    5.1974
5.1864    5.1749    5.1630    5.1506    5.1379    5.1248    5.1114    5.0976
5.0835    5.0692    5.0546    5.0397    5.0246    5.0093    4.9938    4.9782
4.9625    4.9466    4.9307    4.9147    4.8987    4.8826    4.8666    4.8506
4.8347    4.8189    4.8032    4.7876    4.7722    4.7570    4.7421    4.7274
4.7129    4.6988    4.6849    4.6715    4.6584    4.6457    4.6334    4.6215
4.6101    4.5992    4.5887    4.5788    4.5694    4.5606    4.5523    4.5445
4.5373    4.5307    4.5247    4.5193    4.5144    4.5101    4.5064    4.5032
4.5005    4.4984    4.4968    4.4956    4.4949    4.4946    4.4946    4.4950
4.4956    4.4964    4.4974];
tibiaangle_desired = [4.2853    4.2933    4.3011    4.3088    4.3164
4.3237    4.3309    4.3380    4.3449    4.3517    4.3584    4.3648    4.3712
4.3774    4.3835    4.3894    4.3952    4.4010    4.4065    4.4120    4.4173
4.4226    4.4277    4.4328    4.4378    4.4426    4.4474    4.4521    4.4566
4.4611    4.4657    4.4701    4.4744    4.4786    4.4828    4.4870    4.4912
4.4952    4.4993    4.5033    4.5073    4.5113    4.5151    4.5191    4.5229
4.5268    4.5307    4.5345    4.5384    4.5422    4.5460    4.5498    4.5537
4.5575    4.5614    4.5653    4.5691    4.5731    4.5769    4.5809    4.5849
4.5889    4.5930    4.5971    4.6011    4.6054    4.6096    4.6139    4.6182
4.6227    4.6272    4.6318    4.6364    4.6412    4.6460    4.6510    4.6561
4.6613    4.6667    4.6723    4.6779    4.6837    4.6899    4.6961    4.7026
4.7093    4.7162    4.7235    4.7310    4.7388    4.7469    4.7554    4.7643
4.7736    4.7833    4.7935    4.8041    4.8153    4.8270    4.8393    4.8393
4.8436    4.8472    4.8499    4.8517    4.8526    4.8523    4.8509    4.8484
4.8449    4.8401    4.8342    4.8272    4.8189    4.8096    4.7991    4.7876
4.7749    4.7613    4.7465    4.7310    4.7145    4.6971    4.6790    4.6600
4.6405    4.6203    4.5995    4.5783    4.5566    4.5345    4.5123    4.4896
4.4669    4.4441    4.4212    4.3983    4.3756    4.3531    4.3308    4.3087
4.2870    4.2657    4.2449    4.2246    4.2049    4.1858    4.1673    4.1496
4.1327    4.1164    4.1011    4.0867    4.0731    4.0604    4.0487    4.0380
4.0282    4.0195    4.0117    4.0050    3.9993    3.9946    3.9910    3.9884
3.9867    3.9861    3.9863    3.9877    3.9899    3.9931    3.9971    4.0020
4.0077    4.0142    4.0213    4.0292    4.0378    4.0471    4.0568    4.0670
4.0777    4.0888    4.1003    4.1122    4.1242    4.1364    4.1487    4.1611
4.1734    4.1858    4.1981    4.2101    4.2220    4.2336    4.2447    4.2556
4.2660    4.2759    4.2853];
%This section plots the final results
theta2 = linspace(0,2*pi,200) ;
[thetaf thetat] = compute_angles(r_opt,theta2);

%These 5 lines shift the final values just as they were shifted to find the
error
[thetaf_min thetaf_min_index] = min(thetaf);
yshift = min(femurangle_desired) - thetaf_min;
thetaf = thetaf + yshift;     thetat = thetat + yshift;
thetat = [thetat(thetaf_min_index:200) thetat(1:thetaf_min_index-1)];
thetaf = [thetaf(thetaf_min_index:200) thetaf(1:thetaf_min_index-1)];
```

```
%Plot the results if real numbers are obtained
if isreal(thetaf)
if isreal(thetat)
    figure(1)
    hold on
    plot(theta2,thetaf,':r',theta2,thetat,'-c')
    plot(theta2,femurangle_desired,'--',theta2,tibiaangle_desired,'-.')
    xlabel('Input Angle, theta2 (rads)')
    ylabel('Output Angles (rads)')
    axis([0 2*pi min(tibiaangle_desired)-.1 max(femurangle_desired)+.1])
    legend('Femur Produced','Tibia Produced','Femur Desired','Tibia
Desired','location','best')
    hold on
end
end

% ------------------------------------------------------------------------
function f = err(r)

femurangle_desired = [4.4974    4.5005    4.5034    4.5063    4.5093
4.5122    4.5152    4.5182    4.5213    4.5245    4.5279    4.5313    4.5349
4.5386    4.5424    4.5464    4.5506    4.5550    4.5595    4.5642    4.5691
4.5742    4.5795    4.5850    4.5907    4.5965    4.6026    4.6089    4.6153
4.6219    4.6288    4.6358    4.6430    4.6503    4.6578    4.6655    4.6734
4.6814    4.6896    4.6979    4.7064    4.7150    4.7237    4.7326    4.7415
4.7506    4.7598    4.7691    4.7785    4.7879    4.7974    4.8070    4.8167
4.8264    4.8362    4.8460    4.8558    4.8657    4.8755    4.8854    4.8953
4.9051    4.9150    4.9248    4.9345    4.9443    4.9539    4.9635    4.9730
4.9825    4.9918    5.0010    5.0101    5.0190    5.0278    5.0364    5.0449
5.0531    5.0611    5.0690    5.0765    5.0838    5.0909    5.0976    5.1040
5.1101    5.1157    5.1211    5.1260    5.1304    5.1344    5.1379    5.1409
5.1433    5.1451    5.1463    5.1468    5.1467    5.1458    5.1441    5.1441
5.1645    5.1831    5.2000    5.2152    5.2289    5.2410    5.2517    5.2610
5.2691    5.2759    5.2815    5.2860    5.2893    5.2917    5.2930    5.2935
5.2929    5.2916    5.2893    5.2863    5.2825    5.2779    5.2727    5.2667
5.2601    5.2529    5.2450    5.2366    5.2276    5.2180    5.2080    5.1974
5.1864    5.1749    5.1630    5.1506    5.1379    5.1248    5.1114    5.0976
5.0835    5.0692    5.0546    5.0397    5.0246    5.0093    4.9938    4.9782
4.9625    4.9466    4.9307    4.9147    4.8987    4.8826    4.8666    4.8506
4.8347    4.8189    4.8032    4.7876    4.7722    4.7570    4.7421    4.7274
4.7129    4.6988    4.6849    4.6715    4.6584    4.6457    4.6334    4.6215
4.6101    4.5992    4.5887    4.5788    4.5694    4.5606    4.5523    4.5445
4.5373    4.5307    4.5247    4.5193    4.5144    4.5101    4.5064    4.5032
4.5005    4.4984    4.4968    4.4956    4.4949    4.4946    4.4946    4.4950
4.4956    4.4964    4.4974];
tibiaangle_desired = [4.2853    4.2933    4.3011    4.3088    4.3164
4.3237    4.3309    4.3380    4.3449    4.3517    4.3584    4.3648    4.3712
4.3774    4.3835    4.3894    4.3952    4.4010    4.4065    4.4120    4.4173
4.4226    4.4277    4.4328    4.4378    4.4426    4.4474    4.4521    4.4566
4.4611    4.4657    4.4701    4.4744    4.4786    4.4828    4.4870    4.4912
4.4952    4.4993    4.5033    4.5073    4.5113    4.5151    4.5191    4.5229
4.5268    4.5307    4.5345    4.5384    4.5422    4.5460    4.5498    4.5537
4.5575    4.5614    4.5653    4.5691    4.5731    4.5769    4.5809    4.5849
4.5889    4.5930    4.5971    4.6011    4.6054    4.6096    4.6139    4.6182
4.6227    4.6272    4.6318    4.6364    4.6412    4.6460    4.6510    4.6561
4.6613    4.6667    4.6723    4.6779    4.6837    4.6899    4.6961    4.7026
```

```
4.7093    4.7162    4.7235    4.7310    4.7388    4.7469    4.7554    4.7643
4.7736    4.7833    4.7935    4.8041    4.8153    4.8270    4.8393    4.8393
4.8436    4.8472    4.8499    4.8517    4.8526    4.8523    4.8509    4.8484
4.8449    4.8401    4.8342    4.8272    4.8189    4.8096    4.7991    4.7876
4.7749    4.7613    4.7465    4.7310    4.7145    4.6971    4.6790    4.6600
4.6405    4.6203    4.5995    4.5783    4.5566    4.5345    4.5123    4.4896
4.4669    4.4441    4.4212    4.3983    4.3756    4.3531    4.3308    4.3087
4.2870    4.2657    4.2449    4.2246    4.2049    4.1858    4.1673    4.1496
4.1327    4.1164    4.1011    4.0867    4.0731    4.0604    4.0487    4.0380
4.0282    4.0195    4.0117    4.0050    3.9993    3.9946    3.9910    3.9884
3.9867    3.9861    3.9863    3.9877    3.9899    3.9931    3.9971    4.0020
4.0077    4.0142    4.0213    4.0292    4.0378    4.0471    4.0568    4.0670
4.0777    4.0888    4.1003    4.1122    4.1242    4.1364    4.1487    4.1611
4.1734    4.1858    4.1981    4.2101    4.2220    4.2336    4.2447    4.2556
4.2660    4.2759    4.2853];
theta2 = linspace(0,2*pi,200);
[thetaf thetat] = compute_angles(r,theta2);

%These 5 lines shift the actual motion to correspond with the desired
%motion
[thetaf_min thetaf_min_index] = min(thetaf);
yshift = min(femurangle_desired) - thetaf_min;
thetaf = thetaf + yshift;     thetat = thetat + yshift;
thetat = [thetat(thetaf_min_index:200) thetat(1:thetaf_min_index-1)];
thetaf = [thetaf(thetaf_min_index:200) thetaf(1:thetaf_min_index-1)];
%plot(theta2,thetat,theta2,thetaf)
e = (abs(thetaf-femurangle_desired).^2 + abs(thetat-
tibiaangle_desired).^2).^2;
% e = (thetaf-femurangle_desired).^2;
% e = (thetat-tibiaangle_desired).^2;
f=sum(e);

% ----------------------------------------------------------------------
function [thetaf thetat] = compute_angles(r,theta2)

r2 = r(1);
r3 = r(2);
r4 = r(3);
rf = r(4);

r1 = 1;
theta1 = 0;

theta4 = -2.*atan((-
2.*r1.*r4.*sin(theta1)+2.*r2.*r4.*sin(theta2)+sqrt(4.*r1.^2.*r2.^2.*cos(theta
2).^2+4.*r1.^2.*r2.^2.*cos(theta1).^2-
4.*r3.^2.*r1.*r2.*cos(theta1).*cos(theta2)-
4.*r3.^2.*r1.*r2.*sin(theta1).*sin(theta2)-
8.*r1.^2.*r2.^2.*cos(theta1).*cos(theta2).*sin(theta1).*sin(theta2)+4.*r1.^3.
*r2.*sin(theta1).*sin(theta2)+4.*r2.^3.*r1.*cos(theta1).*cos(theta2)+4.*r2.^3
.*r1.*sin(theta1).*sin(theta2)-
8.*r1.^2.*r2.^2.*cos(theta1).^2.*cos(theta2).^2-
4.*r1.*r4.^2.*sin(theta1).*r2.*sin(theta2)-r1.^4-r2.^4-r4.^4-
r3.^4+4.*r1.^3.*r2.*cos(theta1).*cos(theta2)-
6.*r1.^2.*r2.^2+2.*r1.^2.*r4.^2+2.*r1.^2.*r3.^2+2.*r2.^2.*r4.^2+2.*r2.^2.*r3.
^2+2.*r4.^2.*r3.^2-4.*r4.^2.*r1.*r2.*cos(theta1).*cos(theta2))))./(-r1.^2-
```

```
r2.^2-
r4.^2+r3.^2+2.*r1.*r2.*cos(theta1).*cos(theta2)+2.*r1.*r2.*sin(theta1).*sin(t
heta2)+2.*r1.*r4.*cos(theta1)-2.*r2.*r4.*cos(theta2)));
thetaf = theta4;

% This part was added for the Tibia
r5 = r(5);
r6 = r(6);
r7 = r(7);
alpha = r(8);

rt = rf;
theta5 = theta2 + alpha;

theta7 = -2.*atan((-2.*r1.*r7.*sin(theta1)-
2.*rf.*r7.*sin(thetaf)+2.*r5.*r7.*sin(theta5)+sqrt(-
8.*r1.^2.*rf.*sin(thetaf).*r5.*sin(theta5).*cos(theta1).^2+8.*r5.^2.*r1.*sin(
theta1).*rf.*sin(thetaf).*cos(theta5).^2-
8.*rf.^2.*r1.*sin(theta1).*r5.*sin(theta5).*cos(thetaf).^2+4.*r5.^2.*rf.^2.*c
os(theta5).^2+4.*r1.^2.*r5.^2.*cos(theta1).^2+4.*r1.^2.*r5.^2.*cos(theta5).^2
+4.*r1.^2.*rf.^2.*cos(theta1).^2+4.*r1.^2.*rf.^2.*cos(thetaf).^2+4.*r5.^2.*rf
.^2.*cos(thetaf).^2-
8.*r1.*cos(theta1).*r5.^2.*cos(theta5).*rf.*sin(thetaf).*sin(theta5)+12.*r1.^
2.*rf.*sin(thetaf).*r5.*sin(theta5)+8.*r1.^2.*cos(theta1).^2.*rf.*cos(thetaf)
.*r5.*cos(theta5)+8.*r1.*cos(theta1).*rf.^2.*cos(thetaf).^2.*r5.*cos(theta5)-
8.*r1.^2.*cos(theta1).*rf.^2.*cos(thetaf).*sin(theta1).*sin(thetaf)+8.*r1.^2.
*cos(theta1).*rf.*cos(thetaf).*sin(theta1).*r5.*sin(theta5)+8.*r1.*cos(theta1
).*rf.^2.*cos(thetaf).*sin(thetaf).*r5.*sin(theta5)-
8.*r1.*cos(theta1).*r5.^2.*cos(theta5).^2.*rf.*cos(thetaf)+8.*r1.^2.*cos(thet
a1).*r5.*cos(theta5).*sin(theta1).*rf.*sin(thetaf)-
8.*r1.^2.*cos(theta1).*r5.^2.*cos(theta5).*sin(theta1).*sin(theta5)-
6.*r1.^2.*r5.^2+2.*r7.^2.*r6.^2+2.*r6.^2.*rf.^2+2.*r7.^2.*rf.^2-r5.^4-r7.^4-
r6.^4-rf.^4-
4.*r1.^3.*cos(theta1).*rf.*cos(thetaf)+4.*r1.^3.*cos(theta1).*r5.*cos(theta5)
-
4.*r1.^3.*sin(theta1).*rf.*sin(thetaf)+4.*r1.^3.*sin(theta1).*r5.*sin(theta5)
+4.*r5.^3.*r1.*cos(theta1).*cos(theta5)+4.*r5.^3.*rf.*cos(thetaf).*cos(theta5
)+4.*r5.^3.*r1.*sin(theta1).*sin(theta5)+4.*r5.^3.*rf.*sin(thetaf).*sin(theta
5)-
4.*rf.^3.*r1.*cos(theta1).*cos(thetaf)+4.*rf.^3.*cos(thetaf).*r5.*cos(theta5)
-
4.*rf.^3.*r1.*sin(theta1).*sin(thetaf)+4.*rf.^3.*sin(thetaf).*r5.*sin(theta5)
-8.*r1.^2.*cos(theta1).^2.*rf.^2.*cos(thetaf).^2-
8.*r1.^2.*cos(theta1).^2.*r5.^2.*cos(theta5).^2-
8.*rf.^2.*cos(thetaf).^2.*r5.^2.*cos(theta5).^2-r1.^4-
4.*r5.^2.*r1.*cos(theta1).*rf.*cos(thetaf)-
12.*r5.^2.*r1.*sin(theta1).*rf.*sin(thetaf)+8.*rf.^2.*cos(thetaf).*r5.*cos(th
eta5).*r1.*sin(theta1).*sin(thetaf)-
8.*rf.*cos(thetaf).*r5.^2.*cos(theta5).*r1.*sin(theta1).*sin(theta5)-
8.*rf.^2.*cos(thetaf).*r5.^2.*cos(theta5).*sin(thetaf).*sin(theta5)-
4.*r7.^2.*r1.*cos(theta1).*r5.*cos(theta5)-
4.*r7.^2.*rf.*cos(thetaf).*r5.*cos(theta5)+4.*r6.^2.*r1.*cos(theta1).*rf.*cos
(thetaf)-4.*r6.^2.*r1.*cos(theta1).*r5.*cos(theta5)-
4.*r6.^2.*rf.*cos(thetaf).*r5.*cos(theta5)+4.*r6.^2.*r1.*sin(theta1).*rf.*sin
(thetaf)-4.*r6.^2.*r1.*sin(theta1).*r5.*sin(theta5)-
4.*r6.^2.*rf.*sin(thetaf).*r5.*sin(theta5)+4.*rf.^2.*r1.*cos(theta1).*r5.*cos
```

```
(theta5)+12.*rf.^2.*r1.*sin(theta1).*r5.*sin(theta5)+4.*r7.^2.*r1.*cos(theta1
).*rf.*cos(thetaf)+4.*r1.*r7.^2.*sin(theta1).*rf.*sin(thetaf)-
4.*r1.*r7.^2.*sin(theta1).*r5.*sin(theta5)-
4.*rf.*r7.^2.*sin(thetaf).*r5.*sin(theta5)+4.*r1.^2.*rf.*cos(thetaf).*r5.*cos
(theta5)+2.*r1.^2.*r7.^2+2.*r1.^2.*r6.^2-
6.*r1.^2.*rf.^2+2.*r5.^2.*r7.^2+2.*r5.^2.*r6.^2-
6.*r5.^2.*rf.^2))./(r1.^2+r5.^2+r7.^2-
r6.^2+rf.^2+2.*r1.*cos(theta1).*rf.*cos(thetaf)-
2.*r1.*cos(theta1).*r5.*cos(theta5)-
2.*rf.*cos(thetaf).*r5.*cos(theta5)+2.*r1.*sin(theta1).*rf.*sin(thetaf)-
2.*r1.*sin(theta1).*r5.*sin(theta5)-
2.*rf.*sin(thetaf).*r5.*sin(theta5)+2.*r1.*r7.*cos(theta1)+2.*rf.*r7.*cos(the
taf)-2.*r5.*r7.*cos(theta5)));
thetat = theta7;
```

Appendix C: Equation for Tibia Angle with Four-Bar Femur.

```
theta7 = -2.*atan((-2.*r1.*r7.*sin(theta1)-
2.*rf.*r7.*sin(thetaf)+2.*r5.*r7.*sin(theta5)+sqrt(-
8.*r1.^2.*rf.*sin(thetaf).*r5.*sin(theta5).*cos(theta1).^2+8.*r5.^2.*r1.*sin(
theta1).*rf.*sin(thetaf).*cos(theta5).^2-
8.*rf.^2.*r1.*sin(theta1).*r5.*sin(theta5).*cos(thetaf).^2+4.*r5.^2.*rf.^2.*c
os(theta5).^2+4.*r1.^2.*r5.^2.*cos(theta1).^2+4.*r1.^2.*r5.^2.*cos(theta5).^2
+4.*r1.^2.*rf.^2.*cos(theta1).^2+4.*r1.^2.*rf.^2.*cos(thetaf).^2+4.*r5.^2.*rf
.^2.*cos(thetaf).^2-
8.*r1.*cos(theta1).*r5.^2.*cos(theta5).*rf.*sin(thetaf).*sin(theta5)+12.*r1.^
2.*rf.*sin(thetaf).*r5.*sin(theta5)+8.*r1.^2.*cos(theta1).^2.*rf.*cos(thetaf)
.*r5.*cos(theta5)+8.*r1.*cos(theta1).*rf.^2.*cos(thetaf).^2.*r5.*cos(theta5)-
8.*r1.^2.*cos(theta1).*rf.^2.*cos(thetaf).*sin(theta1).*sin(thetaf)+8.*r1.^2.
*cos(theta1).*rf.*cos(thetaf).*sin(theta1).*r5.*sin(theta5)+8.*r1.*cos(theta1
).*rf.^2.*cos(thetaf).*sin(thetaf).*r5.*sin(theta5)-
8.*r1.*cos(theta1).*r5.^2.*cos(theta5).^2.*rf.*cos(thetaf)+8.*r1.^2.*cos(thet
a1).*r5.*cos(theta5).*sin(theta1).*rf.*sin(thetaf)-
8.*r1.^2.*cos(theta1).*r5.^2.*cos(theta5).*sin(theta1).*sin(theta5)-
6.*r1.^2.*r5.^2+2.*r7.^2.*r6.^2+2.*r6.^2.*rf.^2+2.*r7.^2.*rf.^2-r5.^4-r7.^4-
r6.^4-rf.^4-
4.*r1.^3.*cos(theta1).*rf.*cos(thetaf)+4.*r1.^3.*cos(theta1).*r5.*cos(theta5)
-
4.*r1.^3.*sin(theta1).*rf.*sin(thetaf)+4.*r1.^3.*sin(theta1).*r5.*sin(theta5)
+4.*r5.^3.*r1.*cos(theta1).*cos(theta5)+4.*r5.^3.*rf.*cos(thetaf).*cos(theta5
)+4.*r5.^3.*r1.*sin(theta1).*sin(theta5)+4.*r5.^3.*rf.*sin(thetaf).*sin(theta
5)-
4.*rf.^3.*r1.*cos(theta1).*cos(thetaf)+4.*rf.^3.*cos(thetaf).*r5.*cos(theta5)
-
4.*rf.^3.*r1.*sin(theta1).*sin(thetaf)+4.*rf.^3.*sin(thetaf).*r5.*sin(theta5)
-8.*r1.^2.*cos(theta1).^2.*rf.^2.*cos(thetaf).^2-
8.*r1.^2.*cos(theta1).^2.*r5.^2.*cos(theta5).^2-
8.*rf.^2.*cos(thetaf).^2.*r5.^2.*cos(theta5).^2-r1.^4-
4.*r5.^2.*r1.*cos(theta1).*rf.*cos(thetaf)-
12.*r5.^2.*r1.*sin(theta1).*rf.*sin(thetaf)+8.*rf.^2.*cos(thetaf).*r5.*cos(th
eta5).*r1.*sin(theta1).*sin(thetaf)-
8.*rf.*cos(thetaf).*r5.^2.*cos(theta5).*r1.*sin(theta1).*sin(theta5)-
8.*rf.^2.*cos(thetaf).*r5.^2.*cos(theta5).*sin(thetaf).*sin(theta5)-
4.*r7.^2.*r1.*cos(theta1).*r5.*cos(theta5)-
4.*r7.^2.*rf.*cos(thetaf).*r5.*cos(theta5)+4.*r6.^2.*r1.*cos(theta1).*rf.*cos
(thetaf)-4.*r6.^2.*r1.*cos(theta1).*r5.*cos(theta5)-
4.*r6.^2.*rf.*cos(thetaf).*r5.*cos(theta5)+4.*r6.^2.*r1.*sin(theta1).*rf.*sin
(thetaf)-4.*r6.^2.*r1.*sin(theta1).*r5.*sin(theta5)-
4.*r6.^2.*rf.*sin(thetaf).*r5.*sin(theta5)+4.*rf.^2.*r1.*cos(theta1).*r5.*cos
(theta5)+12.*rf.^2.*r1.*sin(theta1).*r5.*sin(theta5)+4.*r7.^2.*r1.*cos(theta1
).*rf.*cos(thetaf)+4.*r1.*r7.^2.*sin(theta1).*rf.*sin(thetaf)-
4.*r1.*r7.^2.*sin(theta1).*r5.*sin(theta5)-
4.*rf.*r7.^2.*sin(thetaf).*r5.*sin(theta5)+4.*r1.^2.*rf.*cos(thetaf).*r5.*cos
(theta5)+2.*r1.^2.*r7.^2+2.*r1.^2.*r6.^2-
6.*r1.^2.*rf.^2+2.*r5.^2.*r7.^2+2.*r5.^2.*r6.^2-
6.*r5.^2.*rf.^2))./(r1.^2+r5.^2+r7.^2-
r6.^2+rf.^2+2.*r1.*cos(theta1).*rf.*cos(thetaf)-
2.*r1.*cos(theta1).*r5.*cos(theta5)-
2.*rf.*cos(thetaf).*r5.*cos(theta5)+2.*r1.*sin(theta1).*rf.*sin(thetaf)-
2.*r1.*sin(theta1).*r5.*sin(theta5)-
```

```
2.*rf.*sin(thetaf).*r5.*sin(theta5)+2.*r1.*r7.*cos(theta1)+2.*rf.*r7.*cos(the
taf)-2.*r5.*r7.*cos(theta5)));
```

Appendix D: Equation of Femur Angle from Six-Bar Femur Linkage.

```
theta7 = -2*atan((2*r7*r8*sin(theta8)-2*r2*r7*sin(theta2)-
2*r5*r7*sin(theta5)-sqrt(4*r2^2*r8^2*cos(theta8)^2-
4*r8^2*r2*r5*cos(theta2).*cos(theta5)+8*r5^2*r8*sin(theta5).*sin(theta8)*r2.*
cos(theta2).*cos(theta5)-
8*r2^2*r5^2.*cos(theta2).*cos(theta5).*sin(theta2).*sin(theta5)-
12*r8^2*r2*r5.*sin(theta2).*sin(theta5)-
4*r6^2*r2*r8.*cos(theta2)*cos(theta8)-4*r6^2*r2*r8.*sin(theta2)*sin(theta8)-
4*r6^2*r5*r8.*cos(theta5)*cos(theta8)-
4*r6^2*r5*r8.*sin(theta5)*sin(theta8)+4*r6^2*r2*r5.*cos(theta2).*cos(theta5)+
4*r6^2*r2*r5.*sin(theta2).*sin(theta5)-
8*r2^2*r8^2.*cos(theta2)*cos(theta8).*sin(theta2)*sin(theta8)-
8*r2*r8^2.*cos(theta2)*cos(theta8)^2*r5.*cos(theta5)-
8*r2*r8^2.*cos(theta2)*cos(theta8)*r5.*sin(theta5)*sin(theta8)+8*r2^2*r8.*cos
(theta2).^2*cos(theta8)*r5.*cos(theta5)+8*r2^2*r8.*cos(theta2)*cos(theta8)*r5
.*sin(theta2).*sin(theta5)-
8*r2*r8^2.*sin(theta2)*sin(theta8)*r5.*cos(theta5)*cos(theta8)+8*r2^2*r8.*sin
(theta2)*sin(theta8)*r5.*cos(theta2).*cos(theta5)-
8*r5^2*r8^2.*cos(theta5)*cos(theta8).*sin(theta5)*sin(theta8)+8*r5^2*r8.*cos(
theta5).^2*cos(theta8)*r2.*cos(theta2)+8*r5^2*r8.*cos(theta5)*cos(theta8)*r2.
*sin(theta2).*sin(theta5)-4*r7^2*r8*sin(theta8)*r2.*sin(theta2)-
4*r7^2*r8*sin(theta8)*r5.*sin(theta5)+4*r2*r7^2.*sin(theta2)*r5.*sin(theta5)+
4*r2^2*r5*r8.*cos(theta5)*cos(theta8)+12*r2^2*r5*r8.*sin(theta5)*sin(theta8)+
4*r5^2*r2*r8.*cos(theta2)*cos(theta8)+12*r5^2*r2*r8.*sin(theta2)*sin(theta8)-
4*r7^2*r2*r8.*cos(theta2)*cos(theta8)-
4*r7^2*r5*r8.*cos(theta5)*cos(theta8)+4*r7^2*r2*r5.*cos(theta2).*cos(theta5)+
4*r2^3*r8.*cos(theta2)*cos(theta8)+4*r2^3*r8.*sin(theta2)*sin(theta8)-
4*r2^3*r5.*cos(theta2).*cos(theta5)-
4*r2^3*r5.*sin(theta2).*sin(theta5)+4*r5^3*r8.*cos(theta5)*cos(theta8)+4*r5^3
*r8.*sin(theta5)*sin(theta8)-4*r5^3*r2.*cos(theta2).*cos(theta5)-
4*r5^3*r2.*sin(theta2).*sin(theta5)+4*r8^3*r2.*cos(theta2)*cos(theta8)+4*r8^3
*r2.*sin(theta2)*sin(theta8)+4*r8^3*r5.*cos(theta5)*cos(theta8)+4*r8^3*r5.*si
n(theta5)*sin(theta8)-8*r2^2*r8^2.*cos(theta2).^2*cos(theta8)^2-
8*r5^2*r8^2.*cos(theta5).^2*cos(theta8)^2-
8*r2^2*r5^2.*cos(theta2).^2.*cos(theta5).^2+2*r2^2*r7^2-
6*r2^2*r8^2+2*r2^2*r6^2+2*r5^2*r7^2-
6*r5^2*r8^2+2*r5^2*r6^2+2*r7^2*r8^2+2*r7^2*r6^2+2*r8^2*r6^2-r5^4-r7^4-r8^4-
r6^4-
6*r2^2*r5^2+4*r2^2*r8^2.*cos(theta2).^2+4*r5^2*r8^2*cos(theta8)^2+4*r5^2*r8^2
.*cos(theta5).^2+4*r2^2*r5^2.*cos(theta2).^2+4*r2^2*r5^2.*cos(theta5).^2-
8*r5^2*r2*r8.*sin(theta2)*sin(theta8).*cos(theta5).^2+8*r8^2*r2*r5.*sin(theta
2).*sin(theta5)*cos(theta8)^2-
8*r2^2*r5*r8.*sin(theta5)*sin(theta8).*cos(theta2).^2-
r2^4))./(r2^2+r5^2+r7^2+r8^2-r6^2-2*r2*r8.*cos(theta2)*cos(theta8)-
2*r8*sin(theta8)*r2.*sin(theta2)-2*r5*r8.*cos(theta5)*cos(theta8)-
2*r8*sin(theta8)*r5.*sin(theta5)+2*r2*r5.*cos(theta2).*cos(theta5)+2*r2.*sin(
theta2)*r5.*sin(theta5)-
2*r7*r8*cos(theta8)+2*r2*r7.*cos(theta2)+2*r5*r7.*cos(theta5)));
```

Appendix E: Equation of Tibia Angle with Six-Bar Femur Linkage.

```
theta12 = 2*atan((2*r8*r12*sin(theta8)+2*r11*r12*sin(theta11)-
2*r9*r12.*sin(theta2-alpha)-sqrt(-4*r11^3*r8*cos(theta8)*cos(theta11)-
4*r8^3*r11*cos(theta8)*cos(theta11)-4*r8^3*r11*sin(theta8).*sin(theta11)-
4*r11^3*r8*sin(theta8).*sin(theta11)-
8*r8^2*r11^2*cos(theta8)^2.*cos(theta11).^2+4*r8^2*r9^2*cos(theta8)^2+4*r11^2
*r9^2.*cos(theta11).^2+4*r8^2*r11^2.*cos(theta11).^2+4*r11^2*r9^2*cos(theta2-
alpha).^2+4*r8^2*r9^2.*cos(theta2-alpha).^2-
8*r8^2*r11*r9*sin(theta11).*sin(theta2-alpha).*cos(theta8)^2-
8*r11^2*r8*r9*sin(theta8).*sin(theta2-
alpha).*cos(theta11).^2+4*r8^2*r11^2*cos(theta8)^2+8*r9^2*r8*r11*sin(theta8)*
sin(theta11).*cos(theta2-alpha).^2-4*r9^2*r8*r11*cos(theta8).*cos(theta11)-
12*r9^2*r8*r11*sin(theta8)*sin(theta11)+4*r10^2*r8*r11*cos(theta8).*cos(theta
11)+4*r10^2*r8*r11*sin(theta8)*sin(theta11)+4*r12^2*r8*r11*cos(theta8).*cos(t
heta11)-8*r8^2*r11^2*cos(theta8).*cos(theta11)*sin(theta8).*sin(theta11)-
r11^4-r10^4-r12^4-6*r8^2*r11^2-r9^4-8*r8*r9^2*cos(theta8).*cos(theta2-
alpha).^2*r11.*cos(theta11)+4*r8*r12^2*sin(theta8)*r11*sin(theta11)-r8^4-
6*r8^2*r9^2+2*r8^2*r10^2+2*r8^2*r12^2-
6*r11^2*r9^2+2*r11^2*r10^2+2*r11^2*r12^2+2*r9^2*r10^2+2*r9^2*r12^2+2*r10^2*r1
2^2+4*r8^3*r9*cos(theta8).*cos(theta2-
alpha)+4*r8^3*r9*sin(theta8).*sin(theta2-
alpha)+4*r11^3*r9.*cos(theta11).*cos(theta2-
alpha)+4*r11^3*r9.*sin(theta11).*sin(theta2-
alpha)+4*r9^3*r8*cos(theta8).*cos(theta2-
alpha)+4*r9^3*r8*sin(theta8).*sin(theta2-
alpha)+4*r9^3*r11.*cos(theta11).*cos(theta2-
alpha)+4*r9^3*r11.*sin(theta11).*sin(theta2-alpha)-
8*r8^2*r9^2*cos(theta8)^2.*cos(theta2-alpha).^2-
8*r11^2*r9^2.*cos(theta11).^2.*cos(theta2-alpha).^2-
4*r10^2*r8*r9*cos(theta8).*cos(theta2-alpha)-
4*r8*r12^2*sin(theta8)*r9.*sin(theta2-alpha)-
4*r11*r12^2.*sin(theta11)*r9.*sin(theta2-
alpha)+4*r8^2*r11*r9.*cos(theta11).*cos(theta2-
alpha)+12*r8^2*r11*r9.*sin(theta11).*sin(theta2-
alpha)+8*r8^2*r11*sin(theta8).*sin(theta11)*r9*cos(theta8).*cos(theta2-
alpha)+8*r8*r11^2*sin(theta8).*sin(theta11)*r9.*cos(theta11).*cos(theta2-
alpha)-8*r8^2*r9^2*cos(theta8).*cos(theta2-alpha)*sin(theta8).*sin(theta2-
alpha)-8*r8*r9^2*cos(theta8).*cos(theta2-
alpha)*r11.*sin(theta11).*sin(theta2-alpha)-
8*r8*r9^2*sin(theta8).*sin(theta2-alpha)*r11.*cos(theta11).*cos(theta2-
alpha)-8*r11^2*r9^2.*cos(theta11).*cos(theta2-
alpha).*sin(theta11).*sin(theta2-
alpha)+4*r11^2*r8*r9*cos(theta8).*cos(theta2-
alpha)+12*r11^2*r8*r9*sin(theta8).*sin(theta2-alpha)-
4*r10^2*r8*r9*sin(theta8).*sin(theta2-alpha)-
4*r10^2*r11*r9.*cos(theta11).*cos(theta2-alpha)-
4*r10^2*r11*r9.*sin(theta11).*sin(theta2-alpha)-
4*r12^2*r8*r9*cos(theta8).*cos(theta2-alpha)-
4*r12^2*r11*r9.*cos(theta11).*cos(theta2-
alpha)+8*r8^2*r11*cos(theta8)^2.*cos(theta11)*r9.*cos(theta2-
alpha)+8*r8^2*r11*cos(theta8).*cos(theta11)*r9*sin(theta8).*sin(theta2-
alpha)+8*r8*r11^2*cos(theta8).*cos(theta11).^2*r9.*cos(theta2-
alpha)+8*r8*r11^2*cos(theta8).*cos(theta11)*r9.*sin(theta11).*sin(theta2-
alpha)))./(r8^2+r11^2+r9^2-
r10^2+r12^2+2*r8*r11*cos(theta8).*cos(theta11)+2*r8*r11*sin(theta8).*sin(thet
a11)-2*r8*r9*cos(theta8).*cos(theta2-alpha)-2*r8*r9*sin(theta8).*sin(theta2-
```

```
alpha)-2*r11*r9.*cos(theta11).*cos(theta2-alpha)-
2*r11*r9.*sin(theta11).*sin(theta2-
alpha)+2*r8*r12*cos(theta8)+2*r11*r12.*cos(theta11)-2*r9*r12.*cos(theta2-
alpha)));
```